Lic. Manuel Torres Remon

EDITORIAL MACRO®

Desarrollo de aplicaciones web con PHP y MYSQL

Temario:

Introducción al MySQL - Integración MySQL con PHP - Mantenimiento de registros
Reportes - Manejo de imágenes - Programación orientada a objetos
Objetos de datos PHP (PDO)

Desarrollo de aplicaciones web con PHP y MySQL

EDITORIAL MACRO®

España - México - Colombia - Chile - Ecuador - Perú - Bolivia - Uruguay - Guatemala - Costa Rica

Desarrollo de aplicaciones web con PHP y MySQL
Autor: Manuel Ángel Torres Remon

© Derechos de autor registrados:
Empresa Editora Macro EIRL

© Derechos de edición, arte gráfico y diagramación reservados:
Empresa Editora Macro EIRL

Coordinación de edición:
Cynthia Arestegui Baca

Diseño de portada:
Alejandro Marcas León

Corrección de estilo:
Hansi Sánchez
Magaly Ramon Quiroz

Diagramación:
Lizbeth R. Eufracio Quispe

Edición a cargo de:
© Empresa Editora Macro EIRL
Av. Paseo de la República N.° 5613 , Miraflores, Lima, Perú

Teléfono: (511) 748 0560
E-mail: proyecto@editorialmacro.com
Página web: www.editorialmacro.com

Primera edición: enero de 2015

Disponible en: macro.bibliotecasenlinea.com

ISBN N.° 978-612-304-255-4
ISBN N.° 978-612-304-382-7

Prohibida la reproducción parcial o total, por cualquier medio o método, de este libro sin previa autorización de la Empresa Editora Macro EIRL.

Manuel Ángel Torres Remon

Manuel Ángel Torres Remon es Licenciado en Informática. Se dedica a la consultoría y docencia de cursos de tecnología desde hace 15 años. Su formación académica la llevó a cabo en el Instituto Manuel Arévalo Cáceres, en la Universidad Alas Peruanas y en la Universidad Peruana del Norte.

En la actualidad, se desempeña como docente en el Instituto Superior Manuel Arévalo Cáceres, Cibertec y Unimaster, de la Universidad Nacional de Ingeniería, en los cuales imparte cursos de programación, base de datos y análisis de sistemas.

Entre sus publicaciones se encuentran *Fundamentos de programación con Visual Basic 2012, Programación orientada a objetos con Visual Basic 2012, Programación Transact con SQL Server 2012* y su última publicación fue dedicada al HTML5 y CSS3.

Para cualquier duda o consulta sobre el material puede escribir al email: manuel.torresr@hotmail.com.

Agradecimientos

Escuché alguna vez la frase “la lectura de un libro enriquece de conocimientos y empobrece la ignorancia”, pero hoy en día un libro impreso es difícil de leer, pues son pocas las personas que cargan uno en mano, estos han sido reemplazados por los dispositivos móviles. Por tal motivo, mi primer agradecimiento es para usted amigo lector, por adquirir este material que simboliza sacrificio, tiempo y esfuerzo en comprobar que los códigos expuestos son los correctos.

Asimismo, deseo agradecer a la gran familia de la Editorial Macro por confiar nuevamente en mi persona para el desarrollo del presente libro: *Desarrollo de aplicaciones web con PHP y MySQL*, que cuenta con un fundamento expresado en el libro *Fundamentos de programación con PHP*, que recomiendo revisar si los códigos expuestos en este material resultan un tanto complejos.

Finalmente, quiero agradecer enormemente a mis pequeñas hijas Ángela Victoria y Fernanda Ximena Torres Lázaro, a mi esposa Luz y a mi querida suegra Victoria Menor Bautista que siempre tiene un comentario bueno respecto a mi persona y ello me motiva de gran manera a seguir adelante.

Dedicatoria

A mis pequeñas Ángela Victoria y Fernanda Ximena,
que son mi fuente de inspiración, y a mi esposa Luz,
por su comprensión en todo lo que me propongo.

Índice

Introducción..... 11

Capítulo 1

Introducción al MySQL

1.1 Introducción 15
1.2 Características 15
1.3 Versiones de MySQL..... 16
1.4 Descargar MySQL 17
1.4.1 Comprobar con WAMP Server..... 17
1.4.2 Comprobar con XAMPP 18
1.5 Manejo de comandos básicos desde MySQL Console..... 19
1.6 MySQL Workbench..... 21
1.6.1 Descargar MySQL Workbench 21
1.6.2 Instalación de MySQL Workbench 22
1.6.3 Pantalla inicial de MySQL Workbench y conexión a MySQL 24
1.7 MySQL-Front 25
1.7.1 Descargar MySQL-Front 25
1.7.2 Instalación de MySQL-Front..... 26
1.7.3 Creando una nueva conexión a MySQL 29
1.8 Sentencias MySQL para el objeto de base de datos..... 30
1.8.1 Creación de una base de datos..... 30
1.8.2 Listar las bases de datos registradas en el servidor 31
1.8.3 Activar la base de datos..... 31
1.8.4 Mostrar la base de datos activa..... 32
1.8.5 Eliminar la base de datos..... 32
1.9 Sentencias MySQL para el objeto tabla..... 33
1.9.1 Motor de almacenamiento..... 33
1.9.2 Tipos de datos 34
1.9.3 Creación de una tabla..... 36
1.9.4 Agregando columna a la tabla 38
1.9.5 Eliminar columnas de una tabla..... 38
1.9.6 Implementación de columnas con valores nulos..... 39
1.9.7 Implementación de columnas con valores por defecto..... 40
1.9.8 Verificar la estructura de una tabla..... 42
1.9.9 Restringir la duplicidad de valores..... 43
1.9.10 Creación de tablas relacionales 44
1.9.11 Asignación de la llave primaria 45
1.9.12 Visualizar la estructura de la tabla desde MySQL-Front y MySQL Workbench 47
1.9.13 Asignación de la llave foránea 50
1.9.14 Implementación de un esquema de base de datos 51
1.10 Sentencia de Manipulación de Datos (SMD)..... 56
1.10.1 Sentencia INSERT..... 56
1.10.2 Sentencia UPDATE..... 58
1.10.3 Sentencia DELETE..... 60

1.10.4 Sentencia TRUNCATE 61
1.11 Consulta básica con la sentencia SELECT 61
1.11.1 Selección de datos sin restricciones 61
1.11.2 Selección de datos limitando las columnas de la tabla 62
1.11.3 Restringir la muestra de datos repetidos 63
1.11.4 Ordenar de forma ascendente o descendente los registros de una tabla 63
1.11.5 Limitar los registros de la consulta 64
1.12 Combinación de tablas con INNER JOIN 64
1.12.1 Combinación de tablas con LEFT JOIN 66
1.12.2 Combinación de tablas con RIGHT JOIN 67
1.13 Recuperación de datos agrupados 68
1.13.1 Cláusula GROUP BY 68
1.13.2 Función agregada COUNT 69
1.13.3 Función agregada SUM 71
1.13.4 Función agregada MAX 72
1.13.5 Función agregada MIN 72
1.13.6 Función agregada AVG 73
1.13.7 Cláusula HAVING 74

Capítulo 2

Integración MySQL con PHP

2.1 Introducción 77
2.2 Conectar MySQL a PHP 77
2.2.1 Función de conexión al servidor MySQL 77
2.2.2 Función de cierre del servidor 79
2.2.3 Función de selección de base de datos desde el servidor 80
2.2.4 Función que ejecuta una sentencia de consulta MySQL en PHP 82
2.2.5 Función que determine el total de registros devueltos por la función mysql_query usando la sentencia SELECT de MySQL 83
2.2.6 Función que muestre los registros de una consulta mysql_query en PHP 85
2.2.7 Casos desarrollados 91
Caso desarrollado 1: Listado de clientes por distrito. 91
Caso desarrollado 2: Listado de boletas por cliente 97
Caso desarrollado 3: Listado de clientes con imágenes 101
Caso desarrollado 4: Listado de clientes paginado con imágenes 105
2.2.8 Manejo de registros usando arreglos de índice 110
2.2.9 Manejo de registros usando arreglo asociativo 112
2.2.10 Manejo de registros usando arreglos asociativos y foreach 115
2.2.11 Manejo de registros usando procedimientos almacenados 117
Caso desarrollado 5: Consulta de productos por código 120

Capítulo 3

Mantenimiento de registros

3.1 Inserción de registros 127
3.1.1 Casos desarrollados 132
Caso desarrollado 1: Registro del nuevo cliente 132
Caso desarrollado 2: Registro del nuevo cliente adjuntando su imagen 142
3.2 Actualización de registros 152
Caso desarrollado: Actualización de datos del cliente 159
3.3 Eliminación de registros 169
Caso desarrollado: Eliminando registro de cliente 173
3.4 Casos desarrollados 184
Caso desarrollado 1: Mantenimiento de registro de clientes 184
Caso desarrollado 2: Mantenimiento de registro de clientes paginado 195

Capítulo 4

Reportes

4.1 Clase FPDF() 209
4.2 Principales funciones de la librería FPDF 210
4.2.1 Objeto FPDF 210
4.2.2 Añadir una nueva página PDF 211
4.2.3 Definición del tipo de letra y tamaño del texto 212
4.2.4 Definición de una celda para un elemento textual 213
4.2.5 Definición de la posición de un elemento textual 214
4.2.6 Definición del color de un elemento 215
4.2.7 Definición de una línea recta 216
4.2.8 Definición de un rectángulo 217
4.2.9 Implementación de una imagen en el documento PDF 218
4.2.10 Definición de la salida de información 218
4.2.11 Casos desarrollados 219
Caso desarrollado 1: Reporte simple de clientes PDF 219
Caso desarrollado 2: Reporte de clientes con banner 221
Caso desarrollado 3: Reporte de clientes con imágenes 224
Caso desarrollado 4: Generación de fotocheck 227
4.3 Clase PHPLOT 232
4.3.1 Tipos de gráficos 232
4.3.2 Caso desarrollado: Generación de gráficos estadísticos 233

Capítulo 5

Manejo de imágenes

5.1 Introducción 239
5.1.1 Casos desarrollados 239
Caso desarrollado 1: Gráfico básico-Listado de clientes por distrito 239
Caso desarrollado 2: Gráfico con PhpLot-Reporte de boletas por mes-Año 2015.... 243
Caso desarrollado 3: Gráfico con PHPLot-Listado de boletas por cliente 248
Caso desarrollado 4: Gráfico con Google Charts-Listado de boletas por cliente 258

Capítulo 6

Programación orientada a objetos

6.1 Introducción 267
6.2 Clases en PHP 267
6.3 Encapsulamiento de datos 270
6.4 Objetos en PHP..... 271
6.5 Método constructor en PHP..... 271
6.6 Casos desarrollados..... 272
Caso desarrollado 1: Clase/objeto-Venta de productos 272
Caso desarrollado 2: Método constructor-Pago de trabajadores..... 280
Caso desarrollado 3: Método constructor-Listado de productos con MySQL 286
Caso desarrollado 4: Mantenimiento de productos orientado a objetos..... 291
Caso desarrollado 5: Mantenimiento de productos con MVC..... 305

Capítulo 7

Objetos de datos PHP (PDO)

7.1 Introducción 331
7.2 Conexión PDO..... 331
7.3 Casos desarrollados..... 332
Caso desarrollado 1: Listado de clientes..... 332
Caso desarrollado 2: Consulta de datos de cliente por código..... 336
Caso desarrollado 3: Listado de clientes por distrito..... 339

Bibliografía 343

Introducción

El libro *Desarrollo de aplicaciones web con PHP y MySQL* le permitirá crear aplicaciones web usando base de datos y, además, le orientará en la programación de objetos.

Para cumplir con dicho objetivo se ha dividido el libro en siete capítulos que son distribuidos de la siguiente manera: el capítulo uno presenta una breve introducción sobre las sentencias MySQL, este capítulo es la base del desarrollo del libro, pues al comprender dichas sentencias podrá generar aplicaciones sin ningún problema de datos. Se sugiere que repase este tema cuando lo crea conveniente y en cualquier capítulo.

El capítulo dos hace referencia a la integración de PHP con MySQL. Se emplean técnicas de conexión al servidor de la base de datos, así como el permiso de uso de la misma. Con ello se aprenderá a mostrar el listado de registros conectados a la base de datos con casos desarrollados y explicados.

En el capítulo tres se aplican los métodos necesarios para un mantenimiento de registros adecuado, como agregar, eliminar o actualizar los datos de una tabla de base de datos, se presentan casos desarrollados y explicados.

Por otra parte, un punto importante dentro de los proyectos web son los informes o reportes que se pueden producir a partir de los datos de una base de datos, tal es así que en el capítulo cuatro se implementan aplicaciones con reportes, usando la clase FPDF.

Otra tarea prioritaria de las aplicaciones web es el manejo de imágenes de forma dinámica, tal es así que el capítulo cinco está dedicado al desarrollo de aplicaciones, se emplean imágenes gráficas referentes a los valores de la base de datos, esto se analizará mediante el uso de las clases PHPLot y Google Charts.

De acuerdo a los supuestos anteriores, se está en la capacidad de crear aplicaciones orientadas a objetos, es decir, se implementará aplicaciones usando la clase y sus objetos a partir de casos desarrollados básicos, como el método constructor, métodos GET/SET y los objetos. Asimismo, se emplea la programación orientada a objetos para conectarnos a una base de datos e implementar aplicaciones web de mantenimiento de registros, usando puras clases.

Finalmente, en el capítulo siete se hace uso de la programación orientada a objetos, mediante la clase PDO de PHP. Además, se muestran casos desarrollados de mantenimiento de registros.

El autor confía en que este libro resulte útil, en particular, a los estudiantes y profesionales para quienes el desarrollo de aplicaciones web con PHP y MySQL sea de su interés.

CAP.

1

Introducción al MySQL

1.1 Introducción

MySQL es considerado como un sistema de gestión de bases de datos relacional, el cual permite administrar los registros, actualización y eliminación de la información tal como lo haría un SQL Server u Oracle.

La característica principal de PHP es el tema de los derechos debido a que cuenta con una licencia dual, por una parte es considerado como *software* libre bajo la licencia GPL (licencia pública general) y por otra es necesaria la compra de la licencia.

Se atribuye el costo de licencia a la empresa sueca MySQL AB desde los primeros meses del 2008, esta a su vez es una subsidiaria de la empresa Sun Microsystems y esta de Oracle Corporation a partir de abril de 2009.

Se considera adquirir la licencia cuando modificamos el código MySQL y queremos redistribuir aplicando un costo por dicha distribución. Dicho de otro modo, lo que es prohibido es distribuir algo que hemos obtenido bajo la licencia GPL; eso quiere decir que podemos usar el servidor MySQL pero no incluirlo dentro del *software* desarrollado, ya que eso distribuiría a MySQL y, por tanto, rompe las reglas GPL; lo que se recomienda es que el *software* que se desarrollará también se distribuya bajo la misma licencia.

A continuación, algunas políticas que se pueden aplicar bajo la licencia GPL:

- Implementar un sistema de información en una organización que solo use al servidor MySQL.
- Se puede modificar el código fuente de MySQL las veces que crea necesaria mientras no distribuya a MySQL con dichos cambios.
- Se permite la distribución de MySQL a pesar de sus modificaciones, pero siempre bajo la licencia GPL.
- Si el sistema de información solo usa a MySQL y no lo distribuye, el sistema puede ser privativo y beneficiarse de él.

Como se mencionó, MySQL fue creada por la empresa “MySQL AB” establecida en Suecia en el año 1995 con el objetivo principal de cumplir con el estándar SQL (Lenguaje de consultas estructurado), sin sacrificar la velocidad, fiabilidad o usabilidad demostrada por otros SGBD.

Se sostiene que el nombre «MySQL» es heredado de su modelo anterior llamado mSQL, que presentaba algunas deficiencias y al cual le incorporaron API para su mejoramiento, denominándolo así MySQL. Otros señalan que las letras iniciales «My» son colocadas en honor a la hija de uno de los creadores. En el caso del logotipo, se sostiene que el nombre del delfín de MySQL es «Sakila» y fue asignado así por los fundadores de la empresa MySQL AB, proponiendo un concurso para dicho nombre llamado *Name the Dolphin*.

Logotipo oficial de MySQL

1.2 Características

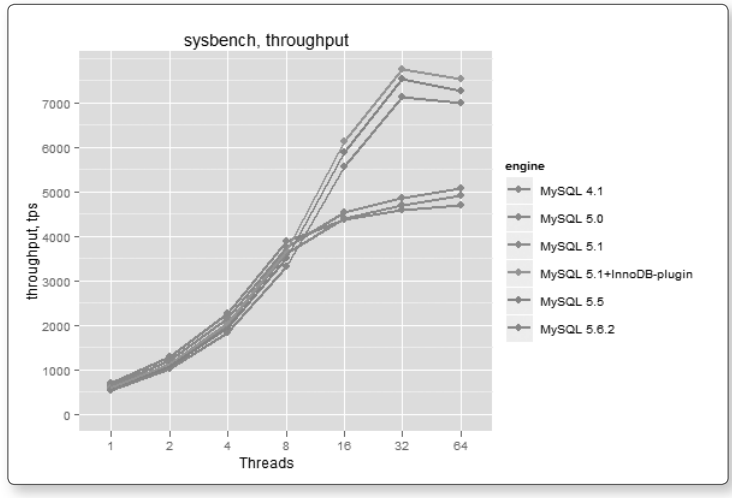
Como ya se había mencionado, uno de los puntos principales de MySQL es la licencia DUAL que mantiene para su uso; pero, además, presenta las siguientes características:

- MySQL fue escrito en su totalidad en el lenguaje C y C++.
- Es considerado como multiplataforma, ya que se puede ejecutar en diferentes sistemas operativos; nosotros lo implementaremos en Windows.

- Es considerado como un sistema portable de licencia GPL ya que usa GNU Automake (herramienta de programación que produce programas makefiles portables usado en la compilación de *software* e implementando especialmente en Linux y Unix), Autoconfig y Libtool.
- Contiene una interfaz de programación de aplicaciones (API) que permite usar MySQL en lenguajes de programación como C, C++, Java, PHP, Perl y otros lenguajes.
- Proporciona un sistema de almacenamiento transaccional y no transaccional con un sistema de reserva de memoria muy rápido basado en hilos y multihilos.
- Las funciones SQL están implementadas usando una librería altamente optimizada y deben ser tan rápidas como sea posible. Normalmente no hay reserva de memoria tras toda la inicialización para consultas.
- El servidor está disponible como un programa separado para usar en un entorno de red cliente/servidor. También está disponible como biblioteca y puede ser incrustado (linkado) en aplicaciones autónomas. Dichas aplicaciones pueden usarse por sí mismas o en entornos donde no hay red disponible.
- Cuenta con un soporte completo para las sentencias SELECT y WHERE, tal como lo maneja SQL Server.
- Cuenta con un sistema de privilegios y contraseñas que es muy flexible y seguro, suficiente para una gestión de la información. Las contraseñas resultan ser totalmente seguras ya que estas son cifradas cuando se conecta con un servidor.
- Cuenta con un soporte importante de almacenamiento de base de datos, se han probado con más 60 000 tablas y cerca de 5 000 000 000 000 de registros.
- Los clientes de MySQL pueden conectarse al servidor usando sockets del TCP/IP.

1.3 Versiones de MySQL

Todas las versiones de MySQL cumplen de manera correcta el objetivo del SGBD, en este material trabajaremos con la versión 5.5 que resulta la versión más estable.



Fuente: <http://www.genbetadev.com/bases-de-datos/mysql-ha-mejorado-el-rendimiento-al-subir-de-version>

1.4 Descargar MySQL

Ingrese a la siguiente URL: **http://dev.mysql.com/downloads/**

De la imagen, seleccione **MySQL Community Server (GPL)**; desde aquí podrá elegir la versión adecuada y especialmente escoger el tipo de sistema del cliente que puede ser 32 o 64 bits.

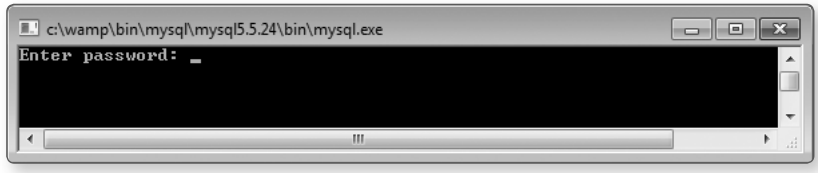
Desde esta ventana se recomienda revisar el tipo de sistema de su computadora personal para poder seleccionar la descarga de MySQL adecuada para su sistema.

1.4.1 Comprobar con WAMP Server

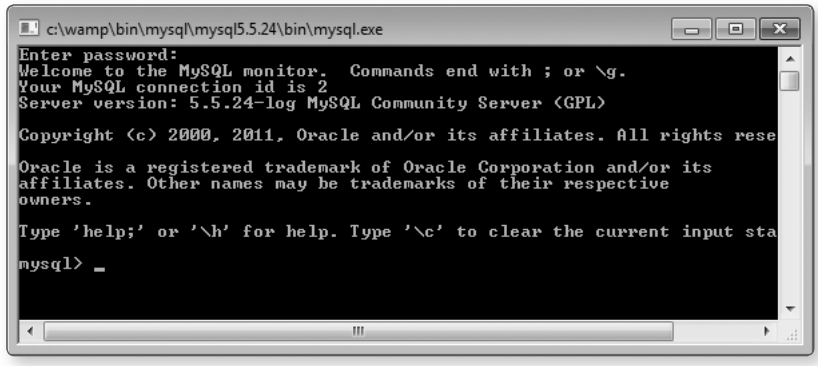
Se debe tener en cuenta que si ha instalado WAMP Server o XAMPP, como servidor de aplicaciones para PHP, entonces ya tiene instalado MySQL. Veamos cómo comprobar con WAMP Server:

- Cargue el servidor WAMP Server.
- Seleccione residente en las aplicaciones de memoria ubicada en la barra de tareas de Windows.

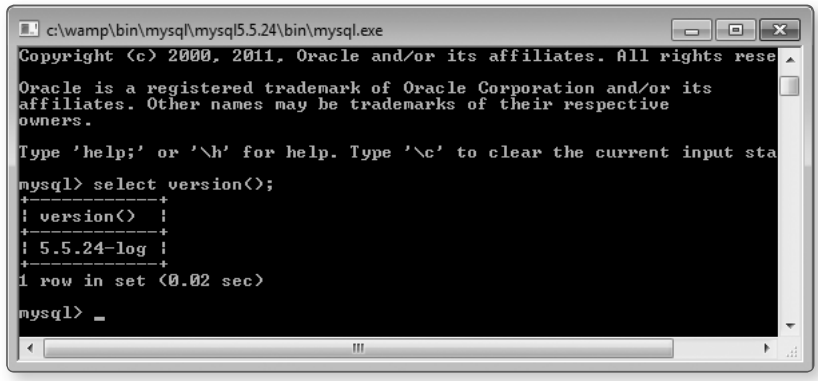
- Seleccione **MySQL > MySQL Console**.



- Desde la ventana anterior, el servidor solicita la clave de acceso, para lo cual solo debe pulsar la tecla **Enter** y visualizará la siguiente ventana:

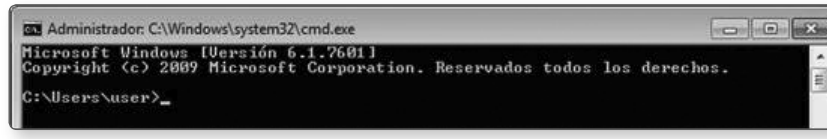


- Para comprobar si todo es correcto, puede ingresar la siguiente instrucción: **select version();**

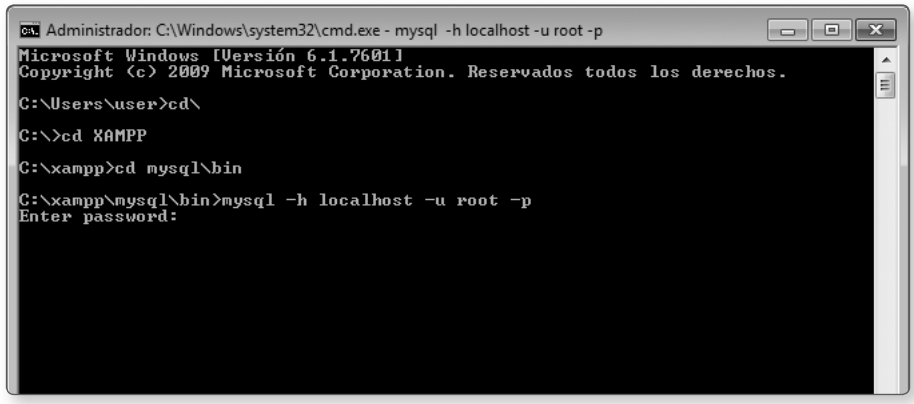


1.4.2 Comprobar con XAMPP

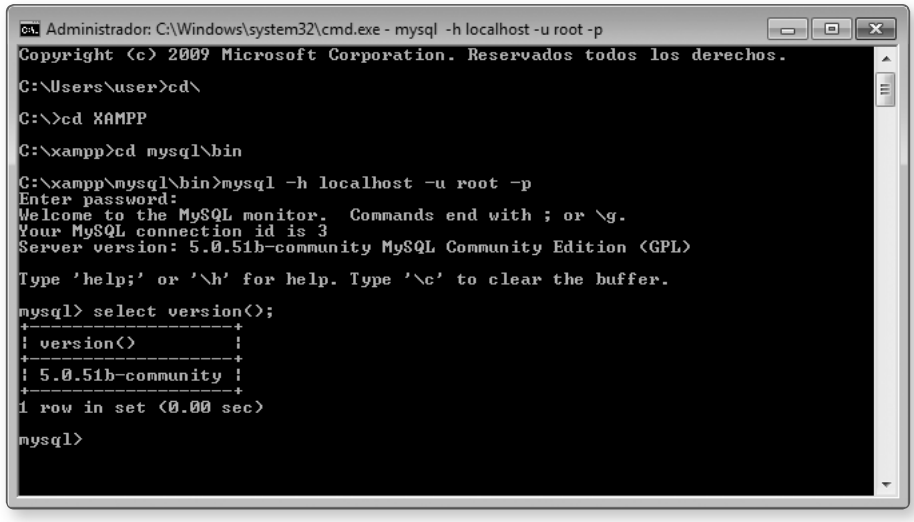
- Asegúrese que otro servidor web no se encuentre ejecutado en nuestra computadora personal.
- Cargue el servidor XAMPP.
- Verifique que en la barra de tareas se encuentre el ícono
- Pulse la combinación de teclas **Windows** + **R** y coloque la instrucción CMD.



- Coloque las siguientes instrucciones en estricto orden para llegar al servidor MySQL.



- Seguidamente, se le solicitará el *password* y, como no se ha configurado, solo pulse la tecla **Enter** y pruebe el servidor con la sentencia: **select version()**;

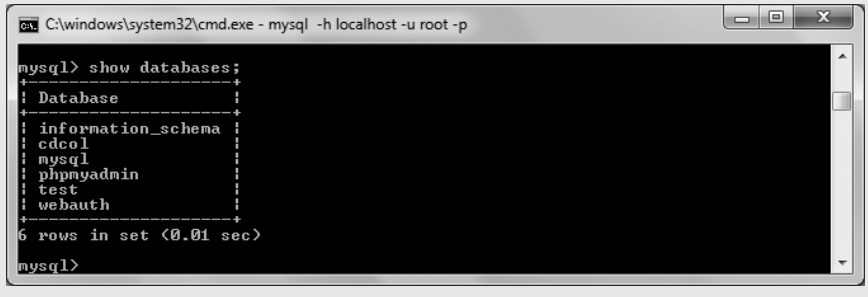


1.5 Manejo de comandos básicos desde MySQL Console

Pruebe algunos comandos MySQL para comprobar que el servidor trabaja de manera correcta:

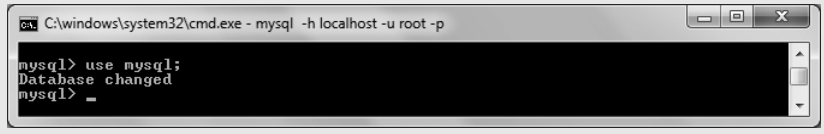
show databases;

Permite mostrar las bases de datos almacenadas en el servidor.



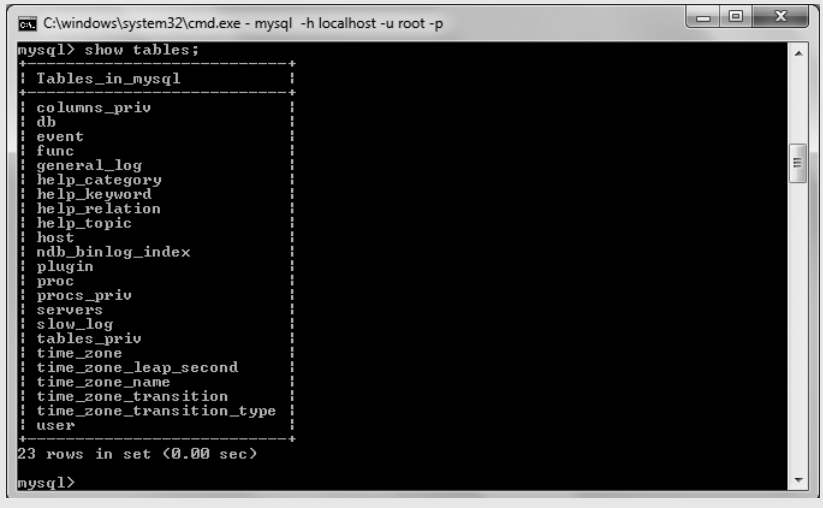
use mysql;

Permite activar una base de datos registrada en el servidor.



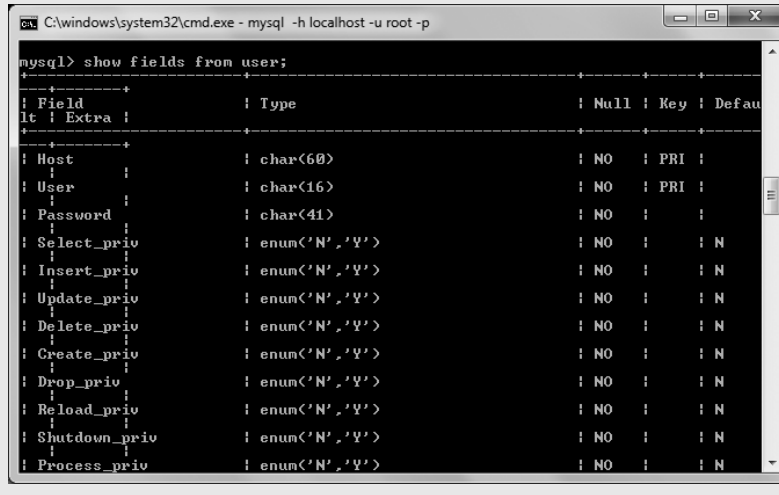
show tables;

Permite listar las tablas de una base de datos activa.



show fields from user;

Permite listar los campos de una tabla, para este caso se ha listado los campos de la tabla **user**.



1.6 MySQL Workbench

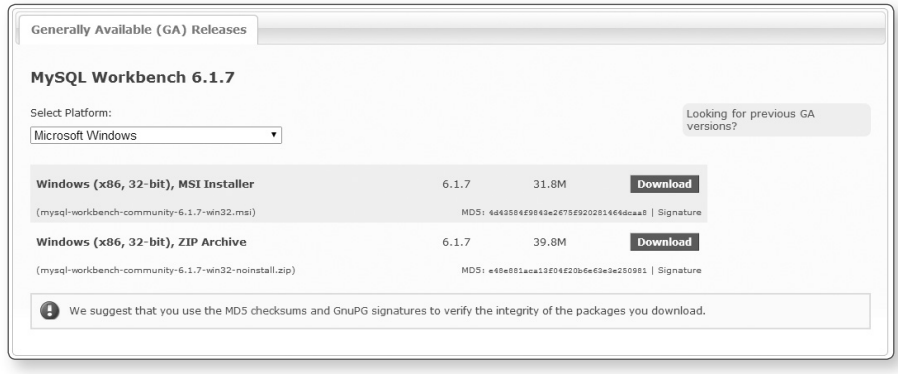
En vista de que sería un poco engorroso trabajar en un entorno tipo D.O.S, se usa una herramienta visual como MySQL Workbench el cual integra la administración de bases de datos, el diseño de bases de datos, creación y mantenimiento para el sistema de base de datos MySQL, todo lo que un sistema de gestión de base de datos necesita.

1.6.1 Descargar MySQL Workbench

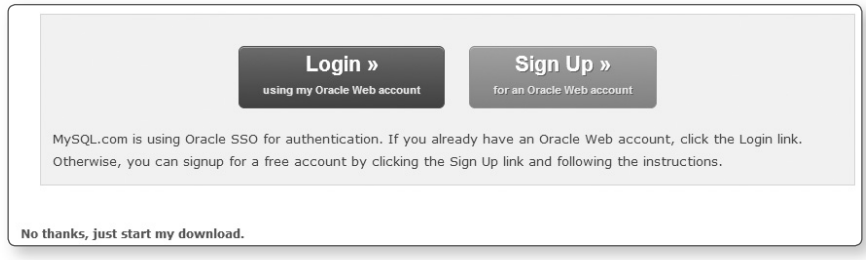
1. Ingrese a la siguiente URL: **http://dev.mysql.com/downloads/**

2. Seleccione **MySQL Workbench.**

3. Seleccione **Download**.



4. Seleccione «No thanks, just start my download».



5. Finalmente, revise la sección de descarga de su computadora personal.

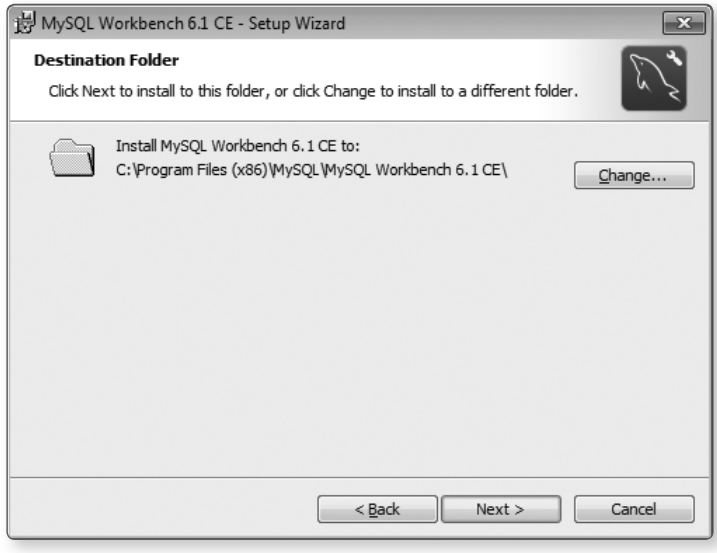
1.6.2 Instalación de MySQL Workbench

Se recuerda que antes de iniciar la instalación de MySQL Workbench debe contar con MySQL instalado en su computadora personal, ya sea usando el propio *software* MySQL o por medio de WAMP Server o XAMPP.

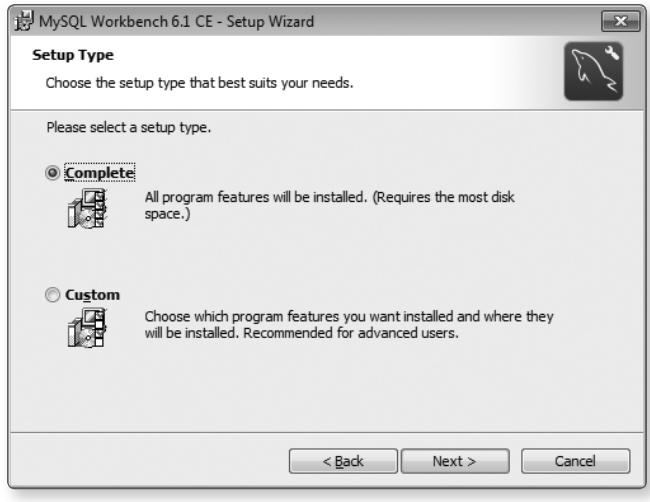
Realice los siguientes pasos:

1. Inicie el proceso de instalación de **MySQL Workbench**.

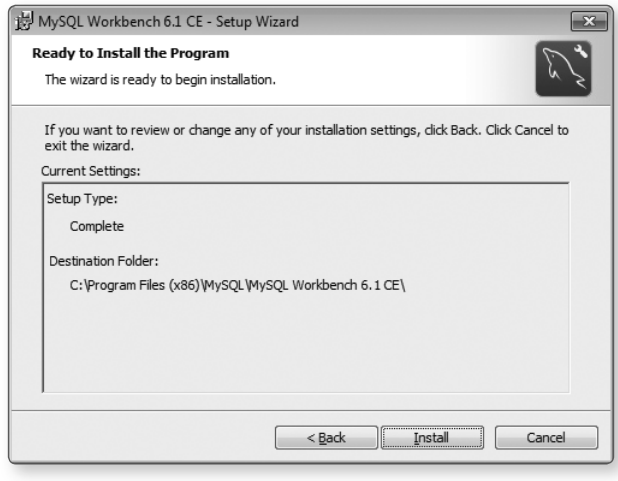
2. No será necesario cambiar la ubicación del lugar de la instalación, así que en el proceso de instalación se procederá a seleccionar **Next** en todas las ventanas.



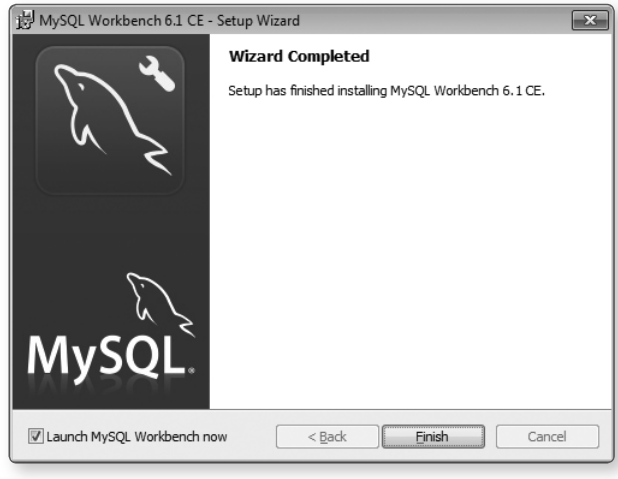
3. En el proceso de instalación, seleccione el tipo **Completo** ya que el *software* viene con todas las especificaciones necesarias para la instalación.



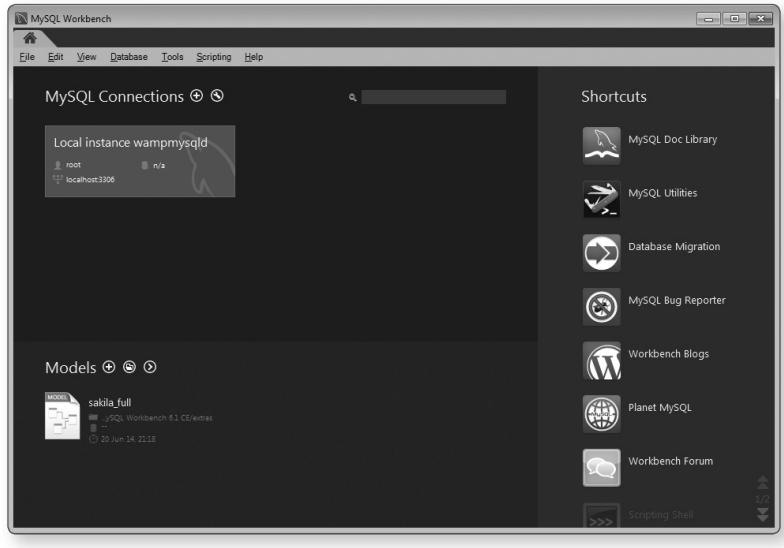
4. Finalmente, inicie la instalación, seleccionando la opción **Install**.



5. La siguiente ventana muestra la finalización de la instalación.



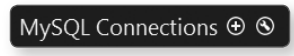
1.6.3 Pantalla inicial de MySQL Workbench y conexión a MySQL

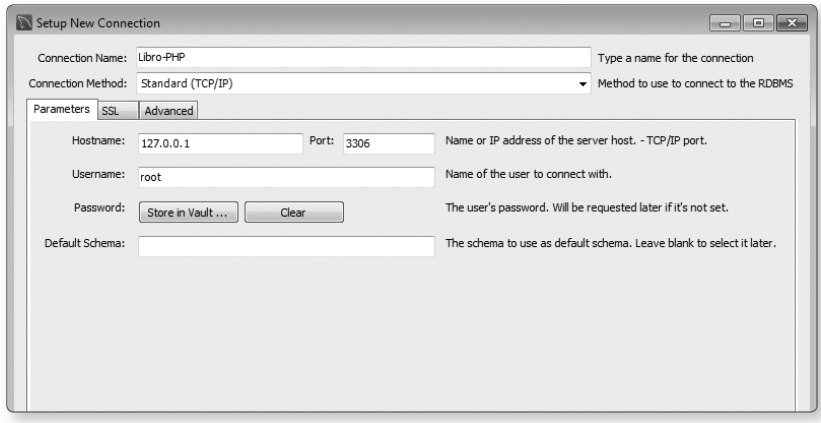


- **Creando una nueva conexión a MySQL**
  Para crear y administrar información de MySQL, se debe crear una nueva conexión que permitirá tener acceso a todos los comandos MySQL.

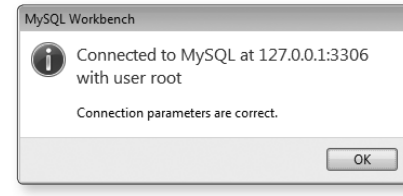
  Realice los siguientes pasos:

1. Desde la ventana principal seleccione la opción **+**.

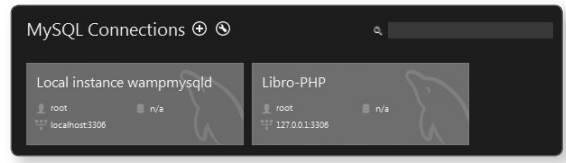




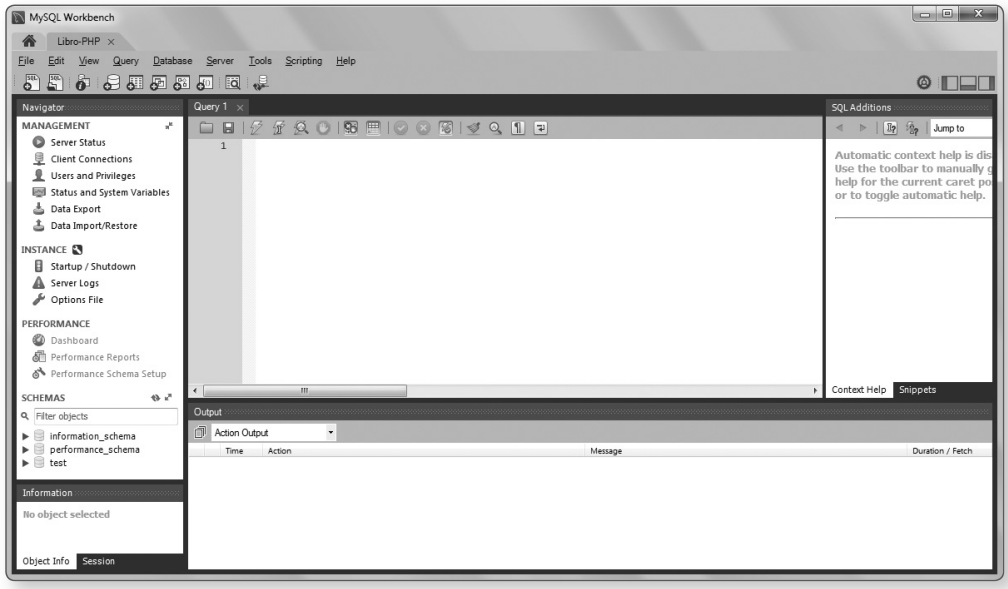
2. Desde la ventana **Setup New Connection** se deberá registrar un nombre a la conexión, verifique que el **Username** sea «root» y, finalmente, seleccione la opción **Test Connection**. Si todo es correcto debe mostrarse la siguiente ventana:



3. Seguidamente, se mostrará en la pantalla principal la conexión realizada tal como se muestra en la siguiente imagen:



4. Para comenzar a trabajar con MySQL, debe seleccionar la conexión realizada y se mostrará la pantalla principal del entorno; desde aquí podrá administrar toda la información concerniente a un proyecto cliente-servidor con MySQL.



1.7 MySQL-Front

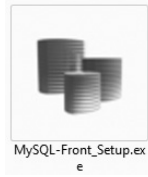
Es una aplicación que permite administrar la información de manera sencilla y es de fácil uso para el usuario de MySQL. Presenta un diseño bastante intuitivo que permite concentrarse en las sentencias de MySQL como una secuencia de comandos así como de manera visual.

1.7.1 Descargar MySQL-Front

1. Ingrese a la siguiente URL: **http://www.mysqlfront.de/**

2. Seleccione la opción **DOWNLOAD FILE**.

3. Finalmente, revise la sección de descarga de su computadora personal.



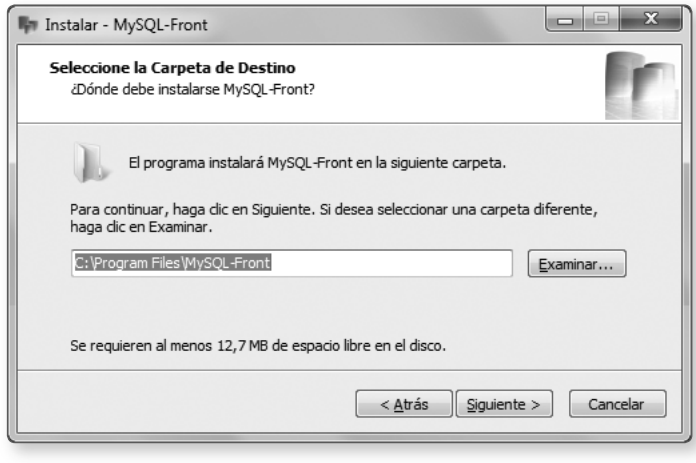
1.7.2 Instalación de MySQL-Front

Se recuerda que antes de iniciar la instalación de MySQL-Front debe contar con MySQL instalado en su computadora personal, ya sea usando el propio *software* MySQL o por medio de WAMP Server o XAMPP.

Realice los siguientes pasos:

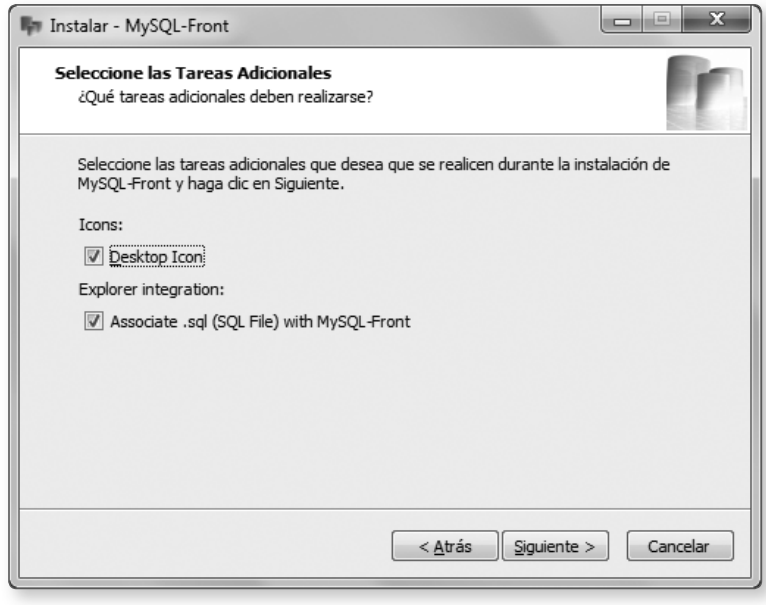
1. Inicie el proceso de instalación de MySQL-Front.

2. No será necesario cambiar la ubicación del lugar de la instalación, así que en el proceso de instalación se procederá a seleccionar **Siguiente** en todas las ventanas.

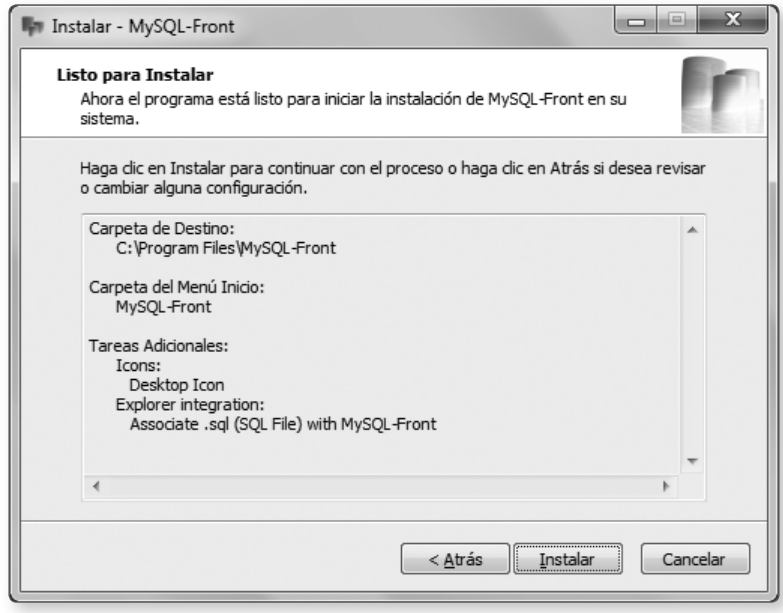


3. La instalación de MySQL-Front crea un acceso directo en el menú Inicio, no será necesario modificar, solo seleccione **Siguiente.**

4. La instalación solicita la creación de un ícono en el escritorio y la asociación con archivos de extensión SQL; los cuales son activados en la siguiente ventana; luego seleccione **Siguiente**.



5. Antes de comenzar la instalación se nos muestra una estadística de opciones seleccionadas en pasos anteriores, solo seleccione **Instalar**.



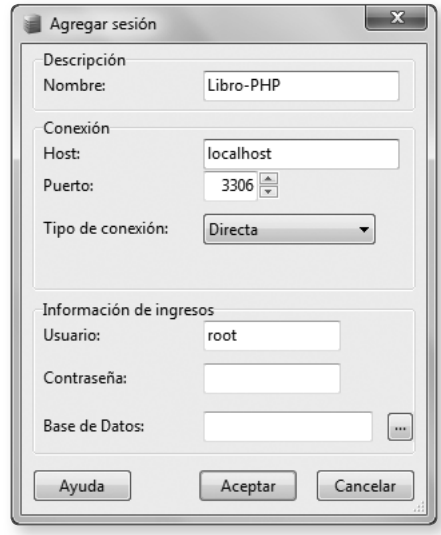
6. Una vez instalado seleccione **Finalizar**.

1.7.3 Creando una nueva conexión a MySQL

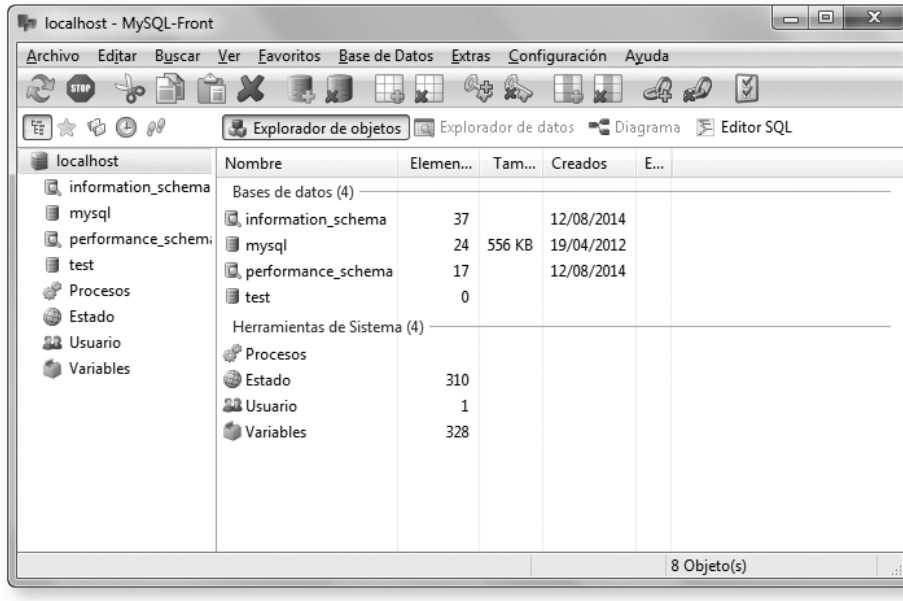
Para crear y administrar información de MySQL se debe crear una nueva conexión que permitirá tener acceso a todos los comandos MySQL.

Realice los siguientes pasos:

- Al iniciar la aplicación MySQL-Front se muestra una ventana de **Agregar sesión**. En esta ventana, en **Nombre** asigne «Libro-PHP», en la opción **Host** asigne «localhost» y asegúrese que en **Usuario** sea «root». No será necesaria la contraseña, por lo tanto, solo seleccione **Aceptar** para iniciar la sesión.



El entorno inicial muestra las bases de datos implementadas en MySQL, así como el Explorador de objetos, Explorador de datos, Diagrama y Editor SQL; este último permitirá ejecutar instrucciones MySQL.



1.8 Sentencias MySQL para el objeto de base de datos

Estas sentencias permiten administrar la información de modo que se puede crear y eliminar una base de datos.

1.8.1 Creación de una base de datos

A. Comentario

La creación de una base de datos permite implementar una base de datos vacía, próximamente debe activarla y, luego, crear tablas y objetos necesarios.

B. Sintaxis

```
CREATE DATABASE [IF NOT EXISTS] NOMBRE_BASE_DATOS
```

- CREATE: Sentencia de definición de datos que permite crear un objeto en el servidor de MySQL.
- DATABASE: Palabra reservada que asociada a la sentencia CREATE permite crear un objeto de base de datos.
- [IF NOT EXISTS]: Cláusula que permite omitir el error que se ocasionaría si la base de datos existiera en el servidor de MySQL.
- NOMBRE_BASE_DATOS: Se especifica el nombre de la base de datos que desea implementar, tener en cuenta que dicho nombre no debe tener espacios en blanco ni símbolos.

Caso 1: Crear la base de datos Ventas de forma básica.

```
create database ventas;
```

Se debe tener en cuenta que si la base de datos Ventas no existe en el servidor, la sentencia se ejecuta correctamente; en caso contrario, emitirá un mensaje de error y no se continuará con la ejecución de las demás sentencias.

Caso 2: Crear la base de datos Ventas validando su existencia, de forma que valide la existencia de la misma.

```
create database IF NOT EXISTS ventas;
```

La base de datos Ventas se creará solo si no existe en el servidor.

Caso 3: Crear la base de datos Ventas validando la existencia de la misma, de forma que si la base de datos existe se deberá eliminar.

```
/* Eliminar la base de datos ventas solo si existe */
drop database if exists ventas;

/* Creando la base de datos ventas */
create database ventas;
```

Al eliminar la base de datos en la primera sentencia, no será necesario validarla en la segunda.

1.8.2 Listar las bases de datos registradas en el servidor

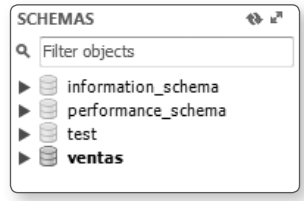
A. Comentario

Una vez creada la base de datos se puede comprobar su existencia haciendo una lista de las bases que contiene el servidor.

B. Sintaxis

```
SHOW DATABASES();
```

Listado de base de datos desde la ventana de Esquemas de MySQL Workbench



1.8.3 Activar la base de datos

A. Comentario

Cuando se crea la base de datos en el servidor de MySQL no se activa de forma automática, esta activación es necesaria para crear, modificar o eliminar objetos dentro de la base de datos.

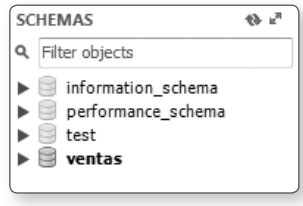
B. Sintaxis

```
USE NOMBRE_BASE_DATOS;
```

Caso 1: Activar la base de datos Ventas.

```
use ventas;
```

Mostrando en MySQL Workbench la base de datos activa, se muestra resaltada en negrita desde la lista de base de datos de la ventana SCHEMAS.



1.8.4 Mostrar la base de datos activa

A. Comentario

Cuando se crean objetos dentro de la base de datos, el servidor ejecuta las sentencias enviadas por el usuario sobre la base de datos activa, por tanto, se debe asegurar que dicha base sea la correcta.

B. Sintaxis

```
SELECT DATABASE();
```

Caso: Mostrar el nombre de la base de datos activa desde el servidor MySQL.

```
USE ventas;
SELECT database();
```

Se debe tener en cuenta que inicialmente se debe activar una base de datos registrada en el servidor, luego se puede comprobar si se encuentra activa o no.

1.8.5 Eliminar la base de datos

A. Comentario

La eliminación de una base de datos del servidor es irreversible, es decir, no podrá recuperar la información almacenada en dicha base de datos.

B. Sintaxis

```
DROP DATABASE NOMBRE_BASE_DATOS;
```

Caso 1: Eliminar la base de datos Ventas de forma estándar.

```
DROP DATABASE ventas;
```

Caso 2: Eliminar la base de datos Ventas condicionando su existencia, es decir, se eliminará si la base de datos Ventas existe; en caso contrario, emitirá un mensaje de advertencia.

```
DROP DATABASE IF EXISTS ventas;
```

Al incluir la cláusula IF EXISTS, se verifica que la base de datos exista, si es así lo eliminará del servidor MySQL; en caso contrario, emitirá un mensaje de advertencia. Ahora supongamos que la base de datos Ventas no existe y se desea eliminarla usando la siguiente sentencia:

```
DROP DATABASE ventas;
```

Se emitirá un error que anulará la ejecución de las sentencias, finalizando así su ejecución. Por lo tanto, es recomendable colocar la cláusula IF EXISTS en la sentencia DROP DATABASE.

1.9 Sentencias MySQL para el objeto tabla

Estas sentencias permiten administrar el objeto de tipo tabla dentro de una base de datos activa. Se debe tener en cuenta que para activar una base de datos se debe usar la sentencia USE.

1.9.1 Motor de almacenamiento

A. Comentario

El motor de almacenamiento tiene la misión de almacenar, administrar y recuperar información contenida en una tabla; actualmente existen muchos motores que realizan las mismas tareas, la diferencia entre ellos es la velocidad y el escenario donde se ejecutará la aplicación. Entre los motores más usados tenemos MyISAM e InnoDB.

- **Motor MyISAM**

Es considerado como el motor por defecto y es comúnmente usado para administrar las consultas de la sentencia SELECT de MySQL.

Para poder asignar el motor MyISAM, se debe seguir el siguiente formato:

```
CREATE TABLE Nombre_de_la_tabla(
     Columna1         tipo,
     Columna2         tipo,
     …..              ….,
     ColumnaN         tipo
) ENGINE=MyISAM;
```

También se debe considerar que cuando no se coloca la cláusula ENGINE se toma por defecto el motor MyISAM. Veamos la implementación de la tabla Cliente que cuenta con las columnas identificador de cliente, nombres, apellidos y su correo electrónico.

```
CREATE TABLE if not exists cliente (
     idCliente        INT,
     nombres          VARCHAR (30),
     apellidos        VARCHAR (30),
     email            VARCHAR (50)
) ENGINE=MyISAM;
```

- **Motor InnoDB**

Se recomienda su aplicación cuando se realice transacciones, ya que estas podrán asegurar su registro con commit y cancelar con rollback. Estas características aumentan el rendimiento para gestionar múltiples usuarios.

```
CREATE TABLE if not exists cliente (
     idCliente        INT,
     nombres          VARCHAR (30),
     apellidos        VARCHAR (30),
     email            VARCHAR (50)
) ENGINE = InnoDB ;
```

1.9.2 Tipos de datos

Entre las partes que componen una tabla de base de datos de MySQL se presenta la especificación del tipo de datos, es decir, qué valor será soportado por dicha columna. Para tomar una decisión del tipo de datos adecuado para cada columna, se recomienda revisar la siguiente tabla:

Numéricos enteros

Bit	Sus valores se encuentran entre 0 y 1. Objetivo: Declarar una variable de bandera que solo acepte los valores 1 y 0. Declaración: bandera bit Valor registrado: 1 o 0
TinyInt	Sus valores se pueden dar en rango de -128 a +127 cuando tiene signo y de 0 a 255 cuando no tiene signo. Objetivo: Declarar una variable para el registro de una nota de un alumno. Declaración: nota tinyint Valor registrado: 20
SmallInt	Sus valores se pueden dar en rango de -32768 a +32767 cuando tiene signo y de 0 a 65535 cuando no tiene signo. Objetivo: Declarar una variable para registrar la cantidad comprada de un determinado producto. Declaración: cantidad smallint Valor registrado: 1500
MediumInt	Sus valores se pueden dar en rango de -8.388.608 a +8.388.607 cuando tiene signo y de 0 a 429.4967.295 cuando no tiene signo. Objetivo: Declarar una variable para registrar la distancia en metros recorridos por un deportista en una carrera de 100 km. Declaración: distancia mediumint Valor registrado: 100000
BigInt	Sus valores se pueden dar en rango de -9.223.372.036.854.775.808 a +9.223.372.036.854.775.807 cuando tiene signo y de 0 a 18.446.744.073.709.551.615 cuando no tiene signo. Objetivo: Declarar una variable para registrar el recorrido de un automóvil entre la ciudad de Lima y Cusco, en metros, si se sabe que tiene una distancia de 1165 km. Declaración: recorrido bigint Valor registrado: 1165000
Integer	Sus valores se pueden dar en rango de -2147483648 a 2147483647 cuando tiene signo y de 0 a 429.4967.295 cuando no tiene signo. Objetivo: Declarar una variable para registrar el total de ventas realizadas por un determinado vendedor en el transcurso de 10 años. Declaración: totalVentas integer Valor registrado: 15206265

Numéricos fraccionarios

Float	Es llamado número pequeño en coma flotante de precisión simple, sus valores tienen el rango de -3.402823466E+38 a -1.175494351E-38, 0 y desde 1.175494351E-38 a 3.402823466E+38. Objetivo: Declarar una variable para registrar la talla de los trabajadores de una empresa de seguridad. Declaración: talla float Valor registrado: 1.85
Double	Es llamado número en coma flotante de precisión doble, sus valores tienen un rango de -1.7976931348623157E+308 a -2.2250738585072014E-308, 0 y desde 2.2250738585072014E-308 a 1.7976931348623157E+308. Objetivo: Declarar una variable para registrar el peso en libras autorizado para camiones de carga de dos ejes si se sabe que dicho valor es 18 toneladas. Declaración: peso double Valor registrado: 41 336.6742
Decimal	Es llamado también número de punto flotante sin empaquetar. Esto se debe a que un valor numérico será registrado como una cadena de caracteres, es decir, un número representará un dígito. El máximo valor almacenado es el mismo que del tipo double. También debemos considerar que tiene analogía con los tipos double(n,d) y numeric(n,d), donde n representa el total de dígitos numéricos, incluido el punto decimal y del número de decimales. Objetivo: Declarar una variable para registrar el sueldo mensual que recibe un trabajador si se sabe que como máximo se puede cobrar el monto de $ 17 000.50. Declaración: sueldoMensual decimal(8,2) Valor registrado: 17 000.50

Cadena de caracteres

Char	Permite registrar caracteres hasta un máximo de 255, asignados entre paréntesis. La longitud mínima es conocida como longitud fija. Si la cantidad asignada es inferior a su máximo límite, este se rellenará de caracteres en blanco hasta completar la especificación del campo. Objetivo: Declarar una variable para registrar el estado civil de un empleado. Declaración: estado_civil char(1) Posibles valores: 'S', 'C', 'V', 'D'
Varchar	Permite registrar una cadena de caracteres con trabajo similar al tipo char, con la diferencia que al definir un tamaño inferior al tamaño máximo no se autorrellena sino se autoajusta al contenido. Objetivo: Declarar una variable para registrar cargo de un empleado en una empresa. Declaración: cargo varchar(80) Valor registrado: 'Operario de planta'
TinyText	Permite registrar una cadena de caracteres con un máximo de 255 caracteres.
Text	Permite registrar una cadena de caracteres con un máximo de 65.535.
MediumText	Permite registrar una cadena de caracteres con un máximo de 16.777.215 caracteres.
LongText	Permite registrar una cadena de caracteres con un máximo de 4.294.967.295 caracteres.

Fecha y hora

Date	Permite registrar una fecha con el siguiente formato 'YYYY-MM-DD' (cuatro dígitos para el año, dos dígitos para el mes, dos dígitos para el día). Su rango es '1000-01-01' hasta '9999-12-31'. Objetivo: Declarar una variable para registrar la fecha de nacimiento del empleado. Declaración: fechaNac date Posibles valores: '1978/10/05'
DateTime	Permite registrar una fecha y hora con el siguiente formato 'YYYY-MM-DD HH:MM:SS' (cuatro dígitos para el año, dos dígitos para el mes, dos dígitos para el día, dos dígitos para la hora, dos dígitos para los minutos, dos últimos dígitos para los segundos). Objetivo: Declarar una variable para registrar la fecha y hora de la impresión de la factura en una venta. Declaración: fechaImpresion dateTime Valor registrado: '2015/10/05 09:00:30'
Time	Permite registrar una hora con el siguiente formato 'HH:MM:SS'. Objetivo: Declarar una variable para registrar la hora de entrada de un empleado. Declaración: horaEntrada time Posibles valores: '09:40:05'
Year	Permite registrar valores numéricos pertenecientes a un año de una fecha, tanto de dos cifras como de cuatro. Objetivo: Declarar una variable que permite registrar el año de ingreso de un empleado a una empresa. Declaración: añoIngreso year Posibles valores: 2013

1.9.3 Creación de una tabla

A. Comentario

Las tablas de una base de datos permiten almacenar información referente a la misma. Se debe tener en cuenta que el nombre que le asigne debe ir acorde con su contenido y que su nombre debe respetar las reglas de los nombres de variables usados en los lenguajes de programación.

B. Sintaxis

```
CREATE TABLE [IF NOT EXISTS] NOMBRE_DE_LA_TABLA (
  COLUMNA TIPO,
  COLUMNA TIPO
);
```

- CREATE: Sentencia de definición de datos que permite crear un objeto en el servidor de MySQL.
- TABLE: Palabra reservada que asociada a la sentencia CREATE permite crear un objeto de tipo tabla en una base de datos activa.
- [IF NOT EXISTS]: Cláusula que permite omitir el error que se ocasionaría si la tabla existiera en la base de datos activa.
- NOMBRE_DE_LA_TABLA: Se especifica el nombre de la tabla que desea implementar, tenga en cuenta que dicho nombre no debe tener espacios en blanco ni símbolos y se recomienda que tenga relación con su contenido.

- COLUMNA: Nombre de los campos de la tabla, tiene la misma regla que el nombre asignado a la tabla.
- TIPO: Es el tipo de datos de cada columna especificada en la implementación de la tabla. Veremos más adelante los tipos de datos especificados en MySQL.

Caso 1: Crear la tabla Producto en la base de datos Ventas de forma básica. La tabla Producto cuenta con las siguientes características:

- Posee un código de cinco caracteres, el cual inicia con la letra P y un número consecutivo de la siguiente forma: P0001.
- La descripción del producto puede tener un máximo de 100 caracteres.
- El precio del producto tiene un máximo rango de $ 999.99.
- El stock del producto representará la cantidad de productos a registrar.
- La fecha de vencimiento del producto se encuentra registrada en el mismo empaque del producto.

```
CREATE TABLE producto(
    idProducto      CHAR(5),
    descripcion     VARCHAR (100),
    precio          DECIMAL(6,2),
    stock           INT,
    fechaVen        DATE
);
```

Caso 2: Crear la tabla Producto validando su existencia en la base de datos Ventas.

```
CREATE TABLE IF NOT EXISTS producto(
    idProducto      CHAR(5),
    descripcion     VARCHAR (100),
    precio          DECIMAL(6,2),
    stock           INT,
    fechaVen        DATE
);
```

Caso 3: Crear la tabla Producto validando la existencia de la misma, de forma que si la tabla existe, lo elimine.

```
/* Activando la base de datos ventas */
USE ventas;

/* Validando la existencia de la tabla Producto */
DROP TABLE IF EXISTS producto;

/* Creando la tabla Producto */
CREATE TABLE producto(
    idProducto      CHAR(5),
    descripcion     VARCHAR (100),
    precio          DECIMAL(6,2),
    stock           INT,
    fechaVen        DATE
);
```

Si es la primera tabla que se creará, se necesita abrir la base de datos con la sentencia USE, luego se crea la tabla Producto, validando primero la existencia de la misma en la base de datos, usando la sentencia DROP TABLE; finalmente, se crea la tabla usando la cláusula IF NOT EXISTS. Se debe tener en cuenta que la tabla Producto ha sido implementada de forma básica, en la cual no ha sido especificada las llaves ni otros valores que componen una tabla.

1.9.4 Agregando columna a la tabla

A. Comentario

Una vez creada la tabla se puede agregar una o más columnas sin necesidad de eliminar la tabla.

B. Sintaxis

```
ALTER TABLE nombre_ tabla
   ADD COLUMN nombre_columna tipo_Datos;
```

Ejemplo
La tabla Producto cuenta con las columnas id_producto, descripción, precio, stock y fecha_venc, tal como se muestra en la siguiente imagen:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
ID_PRODUCTO	char(5)	NO		NULL	
DESCRIPCIÓN	varchar(50)	NO		NULL	
PRECIO	decimal(8,2)	NO		NULL	
STOCK	int(11)	NO		NULL	
FECHA_VENC	date	YES		NULL	

Se agregará la columna ESTADO de tipo varchar(20) con ingreso de valores nulos.

```
/* Agregando la columna estado */
ALTER TABLE producto
     ADD COLUMN estado varchar(20) NULL;

/* Mostrando la estructura de la tabla Producto */
DESCRIBE producto;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
ID_PRODUCTO	char(5)	NO		NULL	
DESCRIPCION	varchar(50)	NO		NULL	
PRECIO	decimal(8,2)	NO		NULL	
STOCK	int(11)	NO		NULL	
FECHA_VENC	date	YES		NULL	
ESTADO	varchar(20)	YES		NULL	

1.9.5 Eliminar columnas de una tabla

A. Comentario

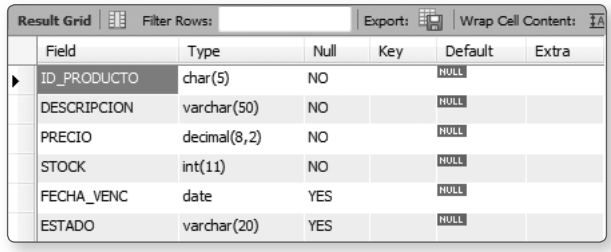
Se puede eliminar una o más columnas alterando la estructura de la tabla sin eliminar por completo dicha tabla.

B. Sintaxis

```
ALTER TABLE nombre_tabla
  DROP COLUMN nombre_columna;
```

Ejemplo

La tabla Producto cuenta con las columnas id_producto, descripción, precio, stock y fecha_venc y estado, tal como se muestra en la siguiente imagen:


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
ID_PRODUCTO	char(5)	NO		NULL	
DESCRIPCION	varchar(50)	NO		NULL	
PRECIO	decimal(8,2)	NO		NULL	
STOCK	int(11)	NO		NULL	
FECHA_VENC	date	YES		NULL	
ESTADO	varchar(20)	YES		NULL	

Eliminar la columna ESTADO de forma que no afecte a las demás columnas de la tabla:

```
ALTER TABLE producto
    DROP COLUMN estado;
```

1.9.6 Implementación de columnas con valores nulos

A. Comentario

Esta definición permite asignar a una columna el registro de un valor nulo, haciendo obligatorio su registro; en caso omita esta implementación, se asumirá que la columna puede contener un valor vacío.

B. Sintaxis

```
CREATE TABLE nombre_tabla (
    Columna1     tipo_de_datos       NOT NULL,
    Columna2     tipo_de_datos       NULL
);
```

Caso 1: Crear la tabla Producto en la base de datos Ventas, la cual cuenta con las siguientes características:

- Cuenta con un código de cinco caracteres, el cual inicia con la letra P y un número consecutivo de la siguiente forma: P0001, además, no debe permitir el registro de valores nulos.
- La descripción del producto puede tener un máximo de 100 caracteres y, además, no debe permitir el registro de valores nulos.
- El precio del producto tiene un máximo rango de $999.99 y, además, no debe permitir el registro de valores nulos.
- El stock del producto representará la cantidad de productos a registrar y, además, no debe permitir el registro de valores nulos.
- La fecha de vencimiento del producto se encuentra registrada en el mismo empaque del producto.

```
/* Validando la existencia de la tabla Producto */
DROP TABLE if exists producto;

/* Creando la tabla Producto */
CREATE TABLE producto(
    ID_PRODUCTO      CHAR(5)          NOT NULL,
    DESCRIPCION      VARCHAR (50)     NOT NULL,
    PRECIO           DECIMAL(8,2)     NOT NULL,
    STOCK            INT              NOT NULL,
    FECHA_VENC       DATE
);
```

Caso 2: Si la tabla Producto ya se encontraba en la base de datos pero de forma básica, es decir, no tenía asignada los valores no nulos, ahora asignaremos los valores NOT NULL a todas las columnas de la tabla Producto.

```
ALTER TABLE producto MODIFY id_Producto char(5) NOT NULL;
ALTER TABLE producto MODIFY descripcion varchar(5) NOT NULL;
ALTER TABLE producto MODIFY precio decimal(6,2) NOT NULL;
ALTER TABLE producto MODIFY stock int NOT NULL;
```

Para verificar las alteraciones realizadas así como la asignación de NOT NULL, se puede usar el siguiente script:

```
DESCRIBE producto;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
ID_PRODUCTO	char(5)	NO		NULL	
DESCRIPCIÓN	varchar(50)	NO		NULL	
PRECIO	decimal(8,2)	NO		NULL	
STOCK	int(11)	NO		NULL	
FECHA_VENC	date	YES		NULL	

1.9.7 Implementación de columnas con valores por defecto

A. Comentario

Permite asignar un valor del mismo tipo de datos a una columna; esto se puede dar cuando el usuario no registre dicho valor, entonces se asignará el valor por defecto al contenido de la columna.

B. Sintaxis

```
CREATE TABLE nombre_tabla (
    Columna tipo_de_datos            DEFAULT 'VALOR'
);
```

Caso 1: Crear la tabla Cliente en la base de datos Ventas, la cual cuenta con las siguientes características:

- Cuenta con un código numérico; además, no debe permitir el registro de valores nulos.
- El nombre del cliente registra tanto los nombres como sus apellidos en un mismo campo, no debe permitir el registro de valores nulos.

- Correo electrónico, el cual registra el nombre del correo y su servidor, además, si el usuario no cuenta con correo electrónico el valor por defecto será «No cuenta».
- Fecha de registro de cliente, en la cual se registrará la fecha en la que el usuario es registrado en el sistema; además, si no se registra la fecha, el valor por defecto es la fecha y hora actual.

```
/* Verificar la existencia de la tabla Cliente */
DROP TABLE if exists CLIENTE;

/* Creando la tabla Cliente */
CREATE TABLE cliente(
    idcliente       int             not null,
    nombres         varchar(70)     not null,
    email           varchar(100)    default 'no cuenta',
    fechareg        timestamp       default localtime
);

/* Verificando la estructura de la tabla Cliente */
DESCRIBE cliente;
```

Field	Type	Null	Key	Default	Extra
idcliente	int(11)	NO		NULL	
nombres	varchar(70)	NO		NULL	
email	varchar(100)	YES		NO CUENTA	
fechareg	timestamp	NO		CURRENT_TIMESTAMP	

```
/* Insertando todos los valores */
INSERT INTO cliente
  VALUES (1,'Angela Torres Lazaro','atorres@hotmail.com','2015/02/05');

/* insertando un cliente sin correo electrónico */
INSERT INTO cliente VALUES (2,'Fernanda Torres Lazaro',default,'2015/02/05');

/* insertando un cliente sin correo electrónico ni fecha de registro */
INSERT INTO cliente VALUES (3,'Luz Lázaro Menor',default,default);

/* Listando los registros de Clientes */
SELECT * FROM cliente;
```

IDCLIENTE	NOMBRES	EMAIL	FECHAREG
1	ANGELA TORRES LAZARO	ATORRES@HOTMAIL.COM	2015-02-05 00:00:00
2	FERNANDA TORRES LAZARO	NO CUENTA	2015-02-05 00:00:00
3	LUZ LAZARO MENOR	NO CUENTA	2014-08-16 11:22:04

Caso 2: Crear la tabla Cliente sin especificar los valores por defecto de las columnas email y fechaReg; para luego modificarlos con la sentencia Alter Table.

```
/* Verificar la existencia de la tabla Cliente */
DROP TABLE if exists CLIENTE;

/* Creando la tabla Cliente */
CREATE TABLE cliente(
    idcliente       int             not null,
    nombres varchar(70)             not null,
```

```
    email           varchar(100),
    fechareg        timestamp
);

/* Modificando la configuración de las columnas por defecto */
ALTER TABLE cliente MODIFY email varchar(100) DEFAULT 'NO CUENTA';
ALTER TABLE cliente MODIFY fechareg timestamp DEFAULT localtime;

/* Verificando la estructura de la tabla Cliente */
DESCRIBE cliente;
```

Field	Type	Null	Key	Default	Extra
idcliente	int(11)	NO		NULL	
nombres	varchar(70)	NO		NULL	
email	varchar(100)	YES		NO CUENTA	
fechareg	timestamp	NO		CURRENT_TIMESTAMP	

```
/* Insertando todos los valores */
INSERT INTO cliente
            VALUES (1,'Angela Torres Lazaro','atorres@hotmail.com','2015/02/05');
/* Insertando un cliente sin correo electrónico */
INSERT INTO cliente VALUES (2,'Fernanda Torres Lazaro',default,'2015/02/05');
/* Insertando un cliente sin correo electrónico ni fecha de registro */
INSERT INTO cliente VALUES (3,'Luz Lázaro Menor',default,default);

/* Listando los registros de Clientes */
SELECT * FROM cliente;
```

1.9.8 Verificar la estructura de una tabla

A. Comentario

La visualización de la estructura nos permite verificar la estructura de una tabla en una base de datos activa.

B. Sintaxis

```
DESCRIBE nombre_tabla;
```

Caso: Ver la estructura de la tabla Cliente, usando la sentencia DESCRIBE cliente.

Field	Type	Null	Key	Default	Extra
idcliente	int(11)	NO		NULL	
nombres	varchar(70)	NO		NULL	
email	varchar(100)	YES		NO CUENTA	
fechareg	timestamp	NO		CURRENT_TIMESTAMP	

La imagen mostrada es obtenida desde MySQL Workbench y muestra el nombre de la columna (Field), el tipo de datos (Type), la definición de los valores nulos (NULL), los campos claves (Key) y los valores por defecto (Default).

1.9.9 Restringir la duplicidad de valores

A. Comentario

En ocasiones, una columna de una tabla puede restringir la duplicidad de los valores. Para que no se registre de otra forma, los valores podrán ser repetidos. Por ejemplo, si nos referimos a la descripción de un producto en una tienda, este valor no puede ser repetido, pero en el caso de su precio sí puede haber duplicidad, ya que dos productos pueden tener el mismo precio.

B. Sintaxis

```
CREATE TABLE nombre_tabla (
    Columna tipo_de_datos           UNIQUE
);
```

Caso 1: Crear la tabla Producto en la base de datos Ventas, la cual cuenta con las siguientes características:

- Un código de cinco caracteres que inicia con la letra P y un número consecutivo de la siguiente forma: P0001; además, no debe permitir el registro de valores nulos.
- La descripción del producto puede tener un máximo de 100 caracteres, no debe permitir el registro de valores nulos y dichos valores deben ser únicos.
- El precio del producto tiene un máximo rango de $ 999.99 y, además, no debe permitir el registro de valores nulos.
- El stock del producto representará la cantidad de productos a registrar y, además, no debe permitir el registro de valores nulos.
- La fecha de vencimiento del producto se encuentra registrada en el mismo empaque del producto y debe tener el valor por defecto al día actual.

```
/* Validando la existencia de la tabla Producto */
DROP TABLE if exists producto;

/* Creando la tabla Producto */
CREATE TABLE producto(
    idProducto      CHAR(5)         NOT NULL,
    descripcion     VARCHAR (100)   NOT NULL UNIQUE,
    precio          DECIMAL(6,2)    NOT NULL,
    stock           INT             NOT NULL,
    fechaReg        timestamp       default localtime
);

/* Insertando productos */
INSERT INTO producto VALUES('p0001','Lavadora 10kg',1500,50,default);
INSERT INTO producto VALUES('p0002','Lavadora 10kg',1500,50,'2015/02/05');

/* Listando los registros de productos */
SELECT * FROM producto;
```

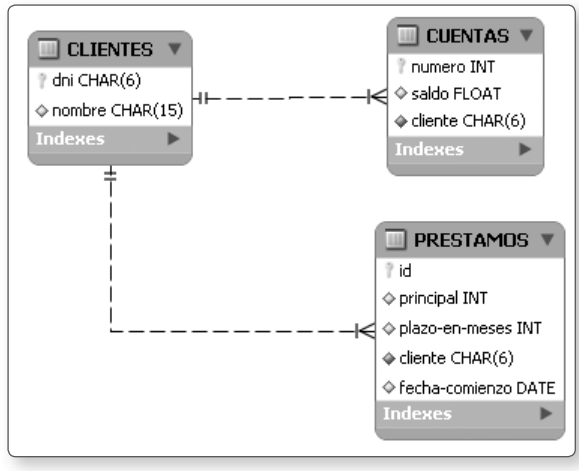
Caso 2: Crear la tabla Producto solo con las cláusulas NOT NULL sin especificar la cláusula UNIQUE a la columna «descripción»; para luego modificarlos con la sentencia Alter Table.

```
/* Validando la existencia de la tabla Producto */
DROP TABLE if exists producto;

/* Creando la tabla Producto */
CREATE TABLE producto(
    idProducto      CHAR(5)         NOT NULL,
    descripcion     VARCHAR (100)   NOT NULL,
    precio          DECIMAL(6,2)    NOT NULL,
    stock           INT             NOT NULL,
    fechaReg        timestamp        default localtime
);

/* Modificando la columna descripción para asignar la no duplicidad */
ALTER TABLE  producto
    MODIFY descripcion VARCHAR (100)         NOT NULL UNIQUE;

/* Insertando productos */
INSERT INTO producto VALUES('p0001','Lavadora 10kg',1500,50,default);
INSERT INTO producto VALUES('p0002','Lavadora 10kg D',1500,50,'2015/02/05');

/* Listando los registros de Clientes */
SELECT * FROM producto;
```

1.9.10 Creación de tablas relacionales

A. Comentario

Las tablas de una base de datos se deben relacionar para evitar que la información sea redundante, es decir, que la información registrada no sea repetida y, por lo tanto, la base de datos sea más pequeña y precisa.



Fuente: <http://cnx.org/contents/39f4b00e-eef5-40e6-a006-52a3ebd97f5a@1>

1.9.11 Asignación de la llave primaria

A. Comentario

La llave primaria, también conocida como principal o simplemente clave, permite definir un identificador único en una tabla. Una o más columnas pueden ser declaradas como llave dentro de una misma tabla. Considere que una columna declarada como campo clave no podrá registrar valores repetidos.

Primera forma: La asignación de llave primaria se realiza directamente sobre la implementación de las columnas.

```
CREATE TABLE nombre_de_la_tabla(
 Columna1    tipo_datos       NOT NULL PRIMARY KEY,
 Columna2    tipo_datos,
 Columna3    tipo_datos

);
```

Segunda forma: La asignación de llave primaria se realiza al finalizar la implementación de las columnas de la tabla.

```
CREATE TABLE nombre_de_la_tabla(
 Columna1    tipo_datos NOT NULL,
 Columna2    tipo_datos,
 Columna3    tipo_datos,
 PRIMARY KEY (Columna1)
);
```

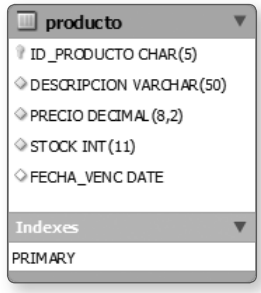
Tercera forma: La asignación de la llave primaria se realiza alterando la tabla, es decir, la tabla se crea inicialmente de forma básica.

```
CREATE TABLE nombre_de_la_tabla(
 Columna1    tipo_datos NOT NULL,
 Columna2    tipo_datos,
 Columna3    tipo_datos
);

ALTER TABLE nombre_de_la_tabla
    ADD PRIMARY KEY (Columna1);
```

Caso: Implementar la tabla Producto dentro de la base de datos Venta con las siguientes características:

- Cuenta con un código de cinco caracteres el cual inicia con la letra P y un número consecutivo de la siguiente forma: P0001; no debe permitir el registro de valores nulos, es la llave primaria de la tabla.
- La descripción del producto puede tener un máximo de 50 caracteres, no debe permitir el registro de valores nulos y dichos valores deben ser únicos.
- El precio del producto tiene un máximo rango de $ 99,999.99 y, además, no debe permitir el registro de valores nulos.

- El stock del producto representará la cantidad de productos a registrar y, además, no debe permitir el registro de valores nulos.
- La fecha de vencimiento del producto se encuentra registrada en el mismo empaque del producto, no es necesario su registro.



La solución se ha dividido en tres versiones, así como los formatos presentados anteriormente:

```
/* PRIMERA VERSIÓN:  Asignación de campo clave directamente */
/* Validando la existencia de la tabla Producto */
DROP TABLE IF EXISTS producto;

/* Creando la tabla Producto */
CREATE TABLE producto(
    ID_PRODUCTO     CHAR(5)         NOT NULL PRIMARY KEY,
    DESCRIPCION     VARCHAR (50)    NOT NULL UNIQUE,
    PRECIO          DECIMAL(8,2)    NOT NULL,
    STOCK           INT             NOT NULL,
    FECHA_VENC      DATE
);

/* Verificando la estructura de la tabla Producto */
DESCRIBE PRODUCTO;

/* SEGUNDO VERSIÓN: Asignación de campo clave al final de la definición de las
columnas */
/* Validando la existencia de la tabla Producto */
DROP TABLE IF EXISTS producto;

/* Creando la tabla Producto */
CREATE TABLE producto(
    ID_PRODUCTO     CHAR(5)         NOT NULL,
    DESCRIPCION     VARCHAR (50)    NOT NULL UNIQUE,
    PRECIO          DECIMAL(8,2)    NOT NULL,
    STOCK           INT             NOT NULL,
    FECHA_VENC      DATE,
    PRIMARY KEY(ID_PRODUCTO)
);

/* Verificando la estructura de la tabla Producto */
DESCRIBE PRODUCTO;

/* TERCERA VERSIÓN: Alterando la tabla para asignar el campo clave */
/* Validando la existencia de la tabla Producto */
DROP TABLE IF EXISTS producto;

/* Creando la tabla Producto */
CREATE TABLE producto(
```

```
    ID_PRODUCTO     CHAR(5)          NOT NULL,
    DESCRIPCION     VARCHAR (50)     NOT NULL UNIQUE,
    PRECIO          DECIMAL(8,2)     NOT NULL,
    STOCK           INT              NOT NULL,
    FECHA_VENC      DATE
);

/* Modificando la tabla Producto */
ALTER TABLE producto
    ADD PRIMARY KEY (ID_PRODUCTO);

/* Verificando la estructura de la tabla Producto */
DESCRIBE PRODUCTO;
```

1.9.12 Visualizar la estructura de la tabla desde MySQL-Front y MySQL Workbench

- **MySQL-Front**

1. Ejecute las siguientes sentencias en el editor de MySQL Front:

```
DROP TABLE IF EXISTS producto;
CREATE TABLE producto(
    ID_PRODUCTO     CHAR(5)          NOT NULL PRIMARY KEY,
    DESCRIPCION     VARCHAR (50)     NOT NULL UNIQUE,
    PRECIO          DECIMAL(8,2)     NOT NULL,
    STOCK           INT              NOT NULL,
    FECHA_VENC      DATE
);
```

2. Desde el menú de opciones de MySQL-Front seleccione **Ver > Diagram.**

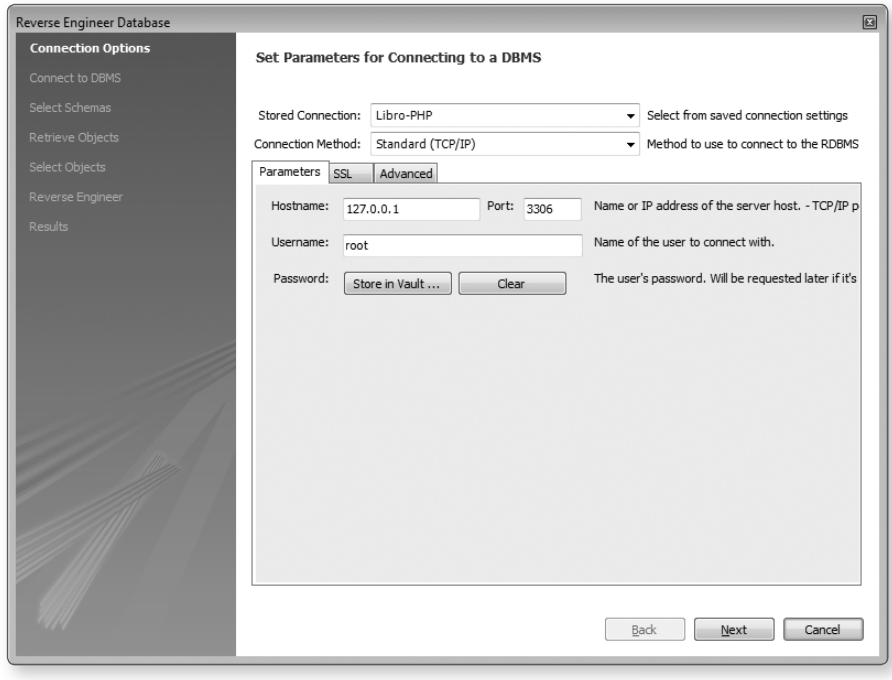
3. Arrastre la tabla desde el explorador (F11).

- **MySQL Workbench**

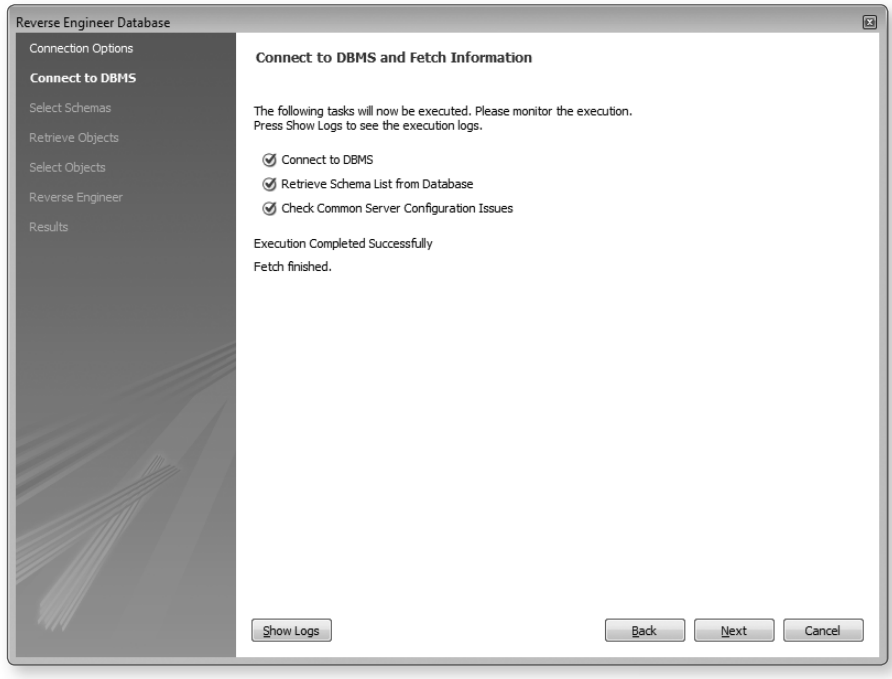
1. Ejecute el siguiente script para la implementación de la tabla:

```
DROP TABLE IF EXISTS producto;
CREATE TABLE producto(
    ID_PRODUCTO     CHAR(5)          NOT NULL PRIMARY KEY,
    DESCRIPCION     VARCHAR (50)     NOT NULL UNIQUE,
    PRECIO          DECIMAL(8,2)     NOT NULL,
    STOCK           INT              NOT NULL,
    FECHA_VENC      DATE
);
```

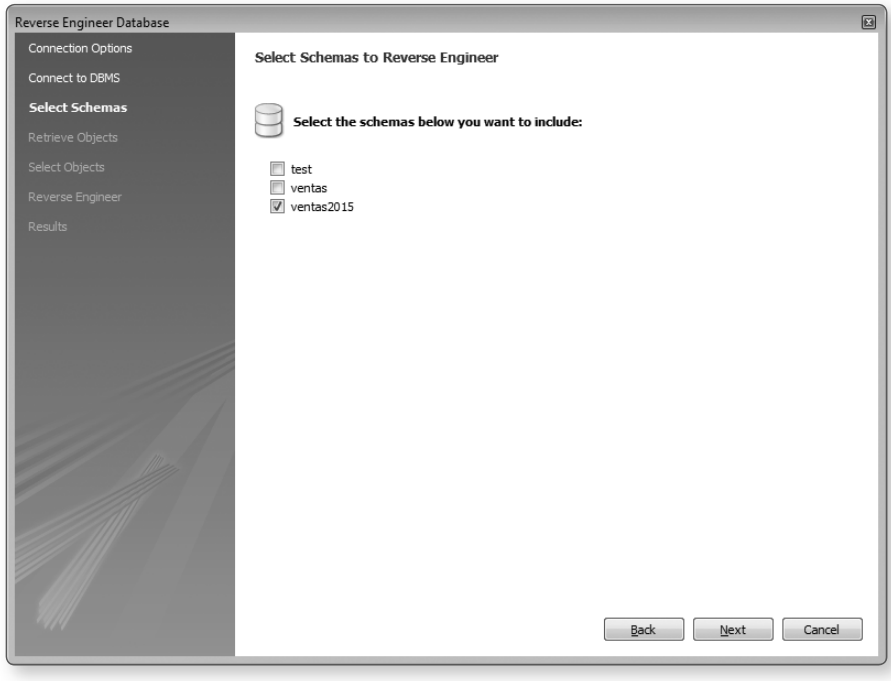
2. Desde el menú de opción seleccione **Database > Reverse Engineer.**. (Ctrl+R). En la ventana solo seleccione **Next**, ya que todo se encuentra configurado.



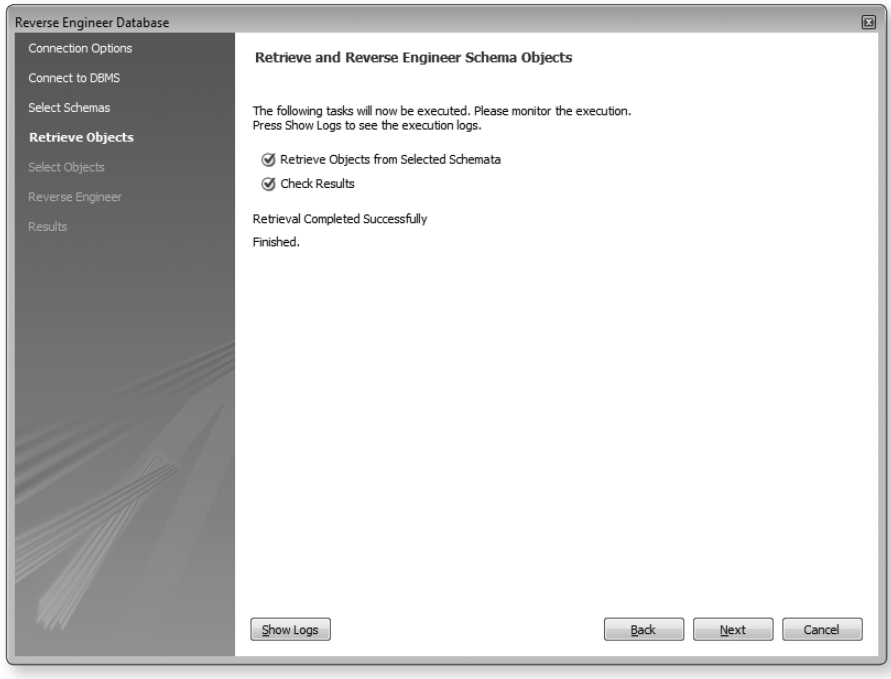
3. La ventana muestra la conexión correcta y la obtención de los datos de la base, aquí solo seleccione **Next**.



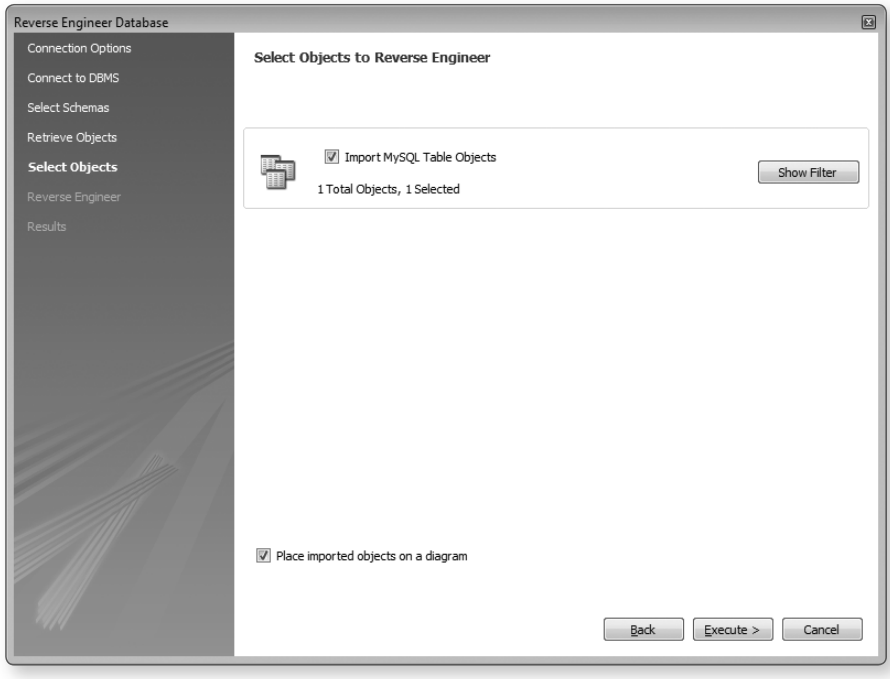
4. En la ventana, active el visto de la base de datos que necesita mostrar su diagrama.

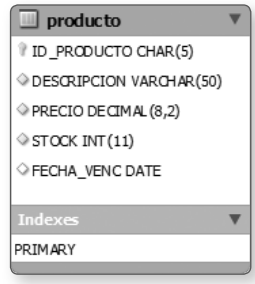


5. En la ventana solo podrá observar que todo el proceso de ingeniería y reingeniería ha sido ejecutado correctamente, seleccione la opción **Next** para mostrar la última ventana del asistente.



6. Para visualizar, seleccione **Execute** para mostrar las tablas asociadas a la base de datos activa.





1.9.13 Asignación de la llave foránea

A. Comentario

La asignación de una llave foránea implementa una base de datos relacional. Normalmente se aplica en una columna de la tabla secundaria. Uno de los requisitos es que tanto la clave primaria como la foránea deben ser del mismo tipo y tamaño.

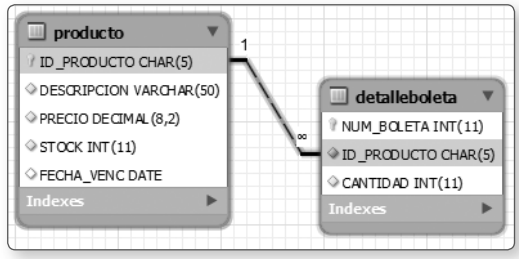
B. Sintaxis

```
ALTER TABLE nombre_de_la_tabla
    ADD FOREIGN KEY (columna_actual)
    REFERENCES tabla_asociada(columna_asociada);
```

Caso: Ver el caso donde se implementa la relación entre la tabla Producto y DetalleBoleta dentro de la base de datos Ventas con las siguientes características:

- La tabla Producto cuenta con las columnas código de producto de 5 caracteres, descripción de 50 caracteres, precio unitario de tipo decimal, stock de tipo entero y la fecha de vencimiento de tipo fecha, siendo el campo clave la columna código del producto.

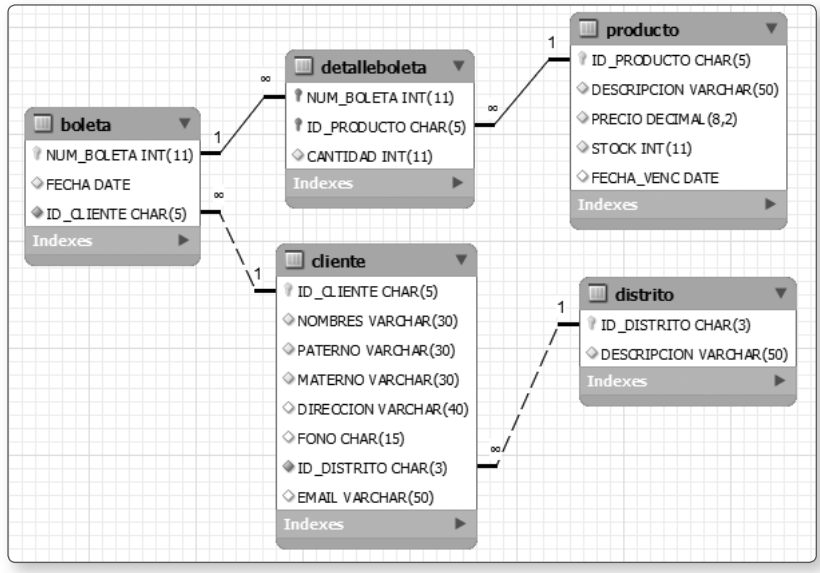
  La tabla DetalleBoleta cuenta con las columnas número de boleta de tipo entero, código del producto de 5 caracteres y la cantidad comprada de tipo entero, tal como se muestra en la siguiente imagen:




```
/* Validando la existencia de la tabla DetalleBoleta */
DROP TABLE IF EXISTS detalleBoleta;

/* Creando la tabla DetalleBoleta */
CREATE TABLE detalleBoleta(
    NUM_BOLETA      INT             NOT NULL PRIMARY KEY,
    ID_PRODUCTO     CHAR(5)         NOT NULL,
    CANTIDAD        INT             NOT NULL
);

/* Agregando la llave foránea a la tabla DetalleBoleta */
ALTER TABLE detalleBoleta
    ADD FOREIGN KEY (id_producto) REFERENCES producto (id_producto);

/* Verificar la estructura de la tabla DetalleBoleta */
DESCRIBE detalleBoleta;
```

1.9.14 Implementación de un esquema de base de datos

Se implementa el modelo relacional de la base de datos Ventas2015, el cual permitirá tener el control de los productos, clientes, distrito de los clientes, boletas y detalles del mismo, la siguiente imagen muestra el diagrama de la base de datos Ventas2015:



Tener en cuenta lo siguiente:

- Se debe asignar las llaves primarias a todas las tablas, tal como se especifica en el diagrama anterior. Solo en el caso de la tabla DetalleBoleta se especificarán dos llaves primarias.
- El número de boleta de la tabla Boleta será un valor numérico autoincrementado que iniciará desde el valor uno.

```
/* Creando la base de datos Ventas2015 */
DROP DATABASE IF EXISTS VENTAS2015;
CREATE DATABASE VENTAS2015;

/* Activando la base de datos Ventas2015 */
USE VENTAS2015;

/*** IMPLEMENTANDO LA TABLA PRODUCTO ***/
/* Validando la existencia de la tabla Producto */
DROP TABLE IF EXISTS producto;

/* Creando la tabla Producto */
CREATE TABLE producto(
    ID_PRODUCTO             CHAR(5)         NOT NULL,
    DESCRIPCION             VARCHAR (50)    NOT NULL,
    PRECIO                  DECIMAL(8,2)    NOT NULL,
    STOCK                   INT             NOT NULL,
    FECHA_VENC              DATE
);

/* Agregando la llave primaria */
ALTER TABLE producto ADD PRIMARY KEY (ID_PRODUCTO);

/*** IMPLEMENTANDO LA TABLA DISTRITO ***/
/* Validando la existencia de la tabla Distrito */
DROP TABLE IF EXISTS distrito;

/* Creando la tabla Distrito */
CREATE TABLE distrito(
```

```
    ID_DISTRITO     CHAR (3)                NOT NULL,
    DESCRIPCION     VARCHAR(50)             NOT NULL
);

/* Agregando la llave primaria */
ALTER TABLE distrito ADD PRIMARY KEY (ID_DISTRITO);

/*** IMPLEMENTANDO LA TABLA CLIENTE ***/
/* Validando la existencia de la tabla Cliente*/
DROP TABLE IF EXISTS cliente;

/* Creando la tabla Cliente */
CREATE TABLE cliente(
    ID_CLIENTE      CHAR (5)        NOT NULL,
    NOMBRES         VARCHAR (30)    NOT NULL,
    PATERNO         VARCHAR (30)    NOT NULL,
    MATERNO         VARCHAR (30)    NOT NULL,
    DIRECCION       VARCHAR (40)    NULL,
    FONO            CHAR (15)       NULL,
    ID_DISTRITO     CHAR (3)        NOT NULL,
    EMAIL           VARCHAR (50)    NULL
);

/* Agregando la llave primaria */
ALTER TABLE cliente ADD PRIMARY KEY (ID_CLIENTE);

/* Agregando la llave foránea a la tabla Cliente */
ALTER TABLE cliente
     ADD FOREIGN KEY (ID_DISTRITO)
     REFERENCES distrito (ID_DISTRITO);

/*** IMPLEMENTANDO LA TABLA BOLETA ***/
/* Validando la existencia de la tabla Boleta*/
DROP TABLE IF EXISTS boleta;

/* Creando la tabla Boleta */
CREATE TABLE boleta(
    NUM_BOLETA              INT             AUTO_INCREMENT,
    FECHA                   DATE            NOT NULL,
    ID_CLIENTE              CHAR (5)        NOT NULL,
    PRIMARY KEY (NUM_BOLETA)
);

/*  Para reinicializar el número autoincrementado
    ALTER TABLE BOLETA AUTO_INCREMENT 1;
*/

/* Agregando la llave foránea a la tabla Boleta */
ALTER TABLE boleta
   ADD FOREIGN KEY (ID_CLIENTE)
   REFERENCES cliente (ID_CLIENTE);

/*** IMPLEMENTANDO LA TABLA DETALLEBOLETA ***/
/* Validando la existencia de la tabla DetalleBoleta */
DROP TABLE IF EXISTS detalleBoleta;

/* Creando la tabla DetalleBoleta */
CREATE TABLE detalleBoleta(
    NUM_BOLETA      INT             NOT NULL,
    ID_PRODUCTO     CHAR(5)         NOT NULL,
    CANTIDAD        INT             NOT NULL
);
```

```
/* Agregando las llaves primarias */
ALTER TABLE detalleBoleta
   ADD PRIMARY KEY (NUM_BOLETA,ID_PRODUCTO);

/* Agregando las llaves foráneas de la tabla DetalleBoleta */
ALTER TABLE detalleBoleta
   ADD FOREIGN KEY (ID_PRODUCTO) REFERENCES producto (ID_PRODUCTO);
ALTER TABLE detalleBoleta
   ADD FOREIGN KEY (NUM_BOLETA) REFERENCES boleta (NUM_BOLETA);

/** INSERTANDO REGISTROS EN LAS TABLAS **/
/* Insertando registros a la tabla Distrito */
INSERT INTO DISTRITO VALUES('L01','CERCADO');
INSERT INTO DISTRITO VALUES('L02','ANCON');
INSERT INTO DISTRITO VALUES('L03','ATE');
INSERT INTO DISTRITO VALUES('L04','BARRANCO');
INSERT INTO DISTRITO VALUES('L05','BREÑA');
INSERT INTO DISTRITO VALUES('L06','CARABAYLLO');
INSERT INTO DISTRITO VALUES('L07','COMAS');
INSERT INTO DISTRITO VALUES('L08','CHACLACAYO');
INSERT INTO DISTRITO VALUES('L09','CHORRILLOS');
INSERT INTO DISTRITO VALUES('L10','EL AGUSTINO');
INSERT INTO DISTRITO VALUES('L11','JESUS MARIA');
INSERT INTO DISTRITO VALUES('L12','LA MOLINA');
INSERT INTO DISTRITO VALUES('L13','LA VICTORIA');
INSERT INTO DISTRITO VALUES('L14','LINCE');
INSERT INTO DISTRITO VALUES('L15','LURIGANCHO');
INSERT INTO DISTRITO VALUES('L16','LURIN');
INSERT INTO DISTRITO VALUES('L17','MAGDALENA');
INSERT INTO DISTRITO VALUES('L18','MIRAFLORES');
INSERT INTO DISTRITO VALUES('L19','PACHACAMAC');
INSERT INTO DISTRITO VALUES('L20','PUCUSANA');

/* Insertando registros a la tabla Productos */
INSERT INTO PRODUCTO VALUES ('P0001','PYE DE MANZANA',20,500,'2015/05/14');
INSERT INTO PRODUCTO VALUES ('P0002','TORTA DE CHOCOLATE',45,100,'2015/07/04');
INSERT INTO PRODUCTO VALUES ('P0003','TORTA TRES LECHES',30,40,'2015/06/24');
INSERT INTO PRODUCTO VALUES ('P0004','MOUSE DE MANZANA',35,70,'2015/09/06');
INSERT INTO PRODUCTO VALUES ('P0005','ARROZ CON LECHE-ENVASE ESPECIAL',13.00,40,'2015/11/04');
INSERT INTO PRODUCTO VALUES ('P0006','MAZAMORRA MORADA',1.50,70,'2015/12/04');
INSERT INTO PRODUCTO VALUES ('P0007','YOGURT ARABE',3,100,'2015/05/04');
INSERT INTO PRODUCTO VALUES ('P0008','PAN CON POLLO',2.00,500,'2015/06/05');
INSERT INTO PRODUCTO VALUES ('P0009','BROWNIE',3,300,'2015/05/04');
INSERT INTO PRODUCTO VALUES ('P0010','BESO DE MOZA',1,400,'2015/06/03');
INSERT INTO PRODUCTO VALUES ('P0011','PYE DE LIMON',1.70,100,'2015/05/02');
INSERT INTO PRODUCTO VALUES ('P0012','TORTA DE NARANJA',16,10,'2015/04/07');
INSERT INTO PRODUCTO VALUES ('P0013','TORTA DE FRESA',41,100,'2015/05/07');
INSERT INTO PRODUCTO VALUES ('P0014','ALFAJORES',0.30,400,'2015/04/06');
INSERT INTO PRODUCTO VALUES ('P0015','CHOCOTEJAS',2,100,'2015/04/14');
INSERT INTO PRODUCTO VALUES ('P0016','SUSPIRO A LA LIMEÑA',3.50,100,'2015/06/12');

/* Insertando registros a la tabla Cliente */
INSERT INTO CLIENTE VALUES ('C0001','CARLOS','LOPEZ', 'HURTADO',
                            'Calle El Pino 346','4677352','L01','clopez@hotmail.com');
INSERT INTO CLIENTE VALUES ('C0002','FRIDA','QUIROZ','DUARTE',
                            'Jr. Los Reyes 475 ','6734732','L06','fquiroz@peru.com');
INSERT INTO CLIENTE VALUES ('C0003','ALEJANDRO','TRAUCO', 'MANRIQUE',
                             'Jr. Huáscar 956','4676732','L04','atruco@gmail.com');
INSERT INTO CLIENTE VALUES ('C0004','CARLA','BLANCO', 'RUIZ',
                             'Av. Arequipa 451','4867438','L03','cblanco@hotmail.com');
INSERT INTO CLIENTE VALUES ('C0005','CORINA','MENDOZA','PEREZ',
```

```
                              'Av. Ingenieros 111','2548965','L09','cmendoza@hotmail.com');
INSERT INTO CLIENTE VALUES ('C0006','JORGE','RODAS', 'DIONICIO',
                              'Jr. Las Liras 456','4789658','L03','jrodas@gmail.com');
INSERT INTO CLIENTE VALUES ('C0007','ORLANDO','CUEVAS', 'CABANILLAS',
                              'Calle La Encantada 425','5698532','L11','ocuevas@peru.com');
INSERT INTO CLIENTE VALUES ('C0008','ANTUANE','RODRIGUEZ','ALARCON',
                              'Av. Dorado 347','4589732','L05','arodriguez@hotmail.com');
INSERT INTO CLIENTE VALUES ('C0009','MANUEL','SUAREZ', 'FERNANDEZ',
                              'Jr. Los Robles 854','4576738','L19','msuarez@gmail.com');
INSERT INTO CLIENTE VALUES ('C0010','JUAN','GUTIERREZ', 'DIAZ',
                              'Calle Girasoles 456','4897421','L14','jgutierrez@hotmail.com');
INSERT INTO CLIENTE VALUES ('C0011','CARLOS','COLAN','BARDALES',
                              'Av. Los Héroes 895','3698574','L19','ccolan@peru.com');
INSERT INTO CLIENTE VALUES ('C0012','MARTIN','CARRILLO', 'SALAS',
                              'Calle Los Huertos 844','8965952','L08','mcarrillo@gmail.com');
INSERT INTO CLIENTE VALUES ('C0013','JOSE','LAZARTE', 'LUJAN',
                              'Jr. Agapito 452','1258965','L17','jlazarte@gmail.com');
INSERT INTO CLIENTE VALUES ('C0014','VIDAL','ZORRILLA','RODRIGUEZ',
                         'Av. Héroes del cenepa 635','1547894','L20','vzorrilla@hotmail.com');
INSERT INTO CLIENTE VALUES ('C0015','GUILLERMO','RAMOS', 'FLORES',
                            'Jr. Las Almendras 211','4587964','L03','gramos@gmail.com');

/* Insertando registros a la tabla Boleta */
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/01/10','C0011');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/01/20','C0002');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/01/20','C0001');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/01/23','C0005');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/02/10','C0001');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/02/20','C0002');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/03/05','C0006');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/03/24','C0006');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/04/20','C0005');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/04/20','C0011');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/05/10','C0006');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/06/20','C0006');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/07/20','C0001');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/07/26','C0006');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/07/20','C0006');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/07/20','C0005');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/08/10','C0002');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/09/05','C0012');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/10/20','C0006');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/11/27','C0014');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/11/05','C0002');
INSERT INTO BOLETA(fecha,id_cliente) VALUES ('2015/11/20','C0001');

/* Insertando registros a la tabla detalleBoleta*/
INSERT INTO DETALLEBOLETA VALUES (1,'P0005',10);
INSERT INTO DETALLEBOLETA VALUES (1,'P0001',20);
INSERT INTO DETALLEBOLETA VALUES (2,'P0002',20);
INSERT INTO DETALLEBOLETA VALUES (2,'P0005',10);
INSERT INTO DETALLEBOLETA VALUES (2,'P0003',15);
INSERT INTO DETALLEBOLETA VALUES (3,'P0005',19);
INSERT INTO DETALLEBOLETA VALUES (4,'P0001',19);
INSERT INTO DETALLEBOLETA VALUES (4,'P0005',15);
INSERT INTO DETALLEBOLETA VALUES (4,'P0003',11);
INSERT INTO DETALLEBOLETA VALUES (5,'P0002',15);
INSERT INTO DETALLEBOLETA VALUES (5,'P0005',10);
INSERT INTO DETALLEBOLETA VALUES (6,'P0003',19);
INSERT INTO DETALLEBOLETA VALUES (7,'P0002',11);
INSERT INTO DETALLEBOLETA VALUES (8,'P0001',10);
```

```
INSERT INTO DETALLEBOLETA VALUES (9,'P0005',11);
INSERT INTO DETALLEBOLETA VALUES (10,'P0015',12);
INSERT INTO DETALLEBOLETA VALUES (11,'P0002',11);
INSERT INTO DETALLEBOLETA VALUES (11,'P0005',15);
INSERT INTO DETALLEBOLETA VALUES (12,'P0003',19);
INSERT INTO DETALLEBOLETA VALUES (12,'P0015',15);
INSERT INTO DETALLEBOLETA VALUES (13,'P0005',11);
INSERT INTO DETALLEBOLETA VALUES (13,'P0002',12);
INSERT INTO DETALLEBOLETA VALUES (14,'P0005',19);
INSERT INTO DETALLEBOLETA VALUES (15,'P0003',12);
INSERT INTO DETALLEBOLETA VALUES (16,'P0015',15);
INSERT INTO DETALLEBOLETA VALUES (17,'P0001',21);
INSERT INTO DETALLEBOLETA VALUES (18,'P0005',21);
INSERT INTO DETALLEBOLETA VALUES (18,'P0004',12);
INSERT INTO DETALLEBOLETA VALUES (19,'P0004',15);
INSERT INTO DETALLEBOLETA VALUES (20,'P0015',30);
INSERT INTO DETALLEBOLETA VALUES (21,'P0005',30);
INSERT INTO DETALLEBOLETA VALUES (22,'P0001',30);

/* Listando todos los registros */
SELECT * FROM CLIENTE;
SELECT * FROM DISTRITO;
SELECT * FROM PRODUCTO;
SELECT * FROM DETALLEBOLETA;
SELECT * FROM BOLETA;
```

1.10 Sentencia de Manipulación de Datos (SMD)

Las sentencias de manipulación corresponden al manejo de todos los datos que se pueden registrar en tablas previamente implementadas.

1.10.1 Sentencia INSERT

A. Comentario

Permite insertar registros en una tabla de una base de datos activa. Se debe considerar los siguientes aspectos:

- Si desea reemplazar la fila registrada, primero debe eliminarlo con la sentencia DELETE o TRUNCATE TABLE; de otra manera se ocasionará un error de duplicidad debido al valor único especificado en la llave primaria.
- Si desea actualizar la fila registrada, debe usar la sentencia UPDATE condicionando el campo clave, ya que es la única forma de identificar a una fila específica.
- Cuando es ejecutada la sentencia INSERT y uno de los registros tiene algún error, el proceso de inserción se detiene y no se ejecutan las demás sentencias.
- Si no especifica un valor en una columna declarada como VARCHAR o TEXT, el motor de base de datos cargará dentro de estas columnas una cadena de longitud 0. Considere también las restricciones implementadas a dichos campos.
- Cuando se insertan valores dentro de una tabla y no se especifica las columnas a registrar, deberá colocar todos los valores obligatorios de la tabla; en caso contrario, se tendrá que especificar las columnas involucradas.

B. Sintaxis

```
INSERT INTO
    TABLA [(Lista de columnas)]
    VALUES ('Lista de Valores')
```

- INTO: Es la palabra clave que se usa entre el INSERT y la especificación de la tabla destino.
- TABLA: Es el nombre de la tabla destino, es decir, la que recibirá los datos.
- Lista de columnas: Esta permite especificar las columnas que tendrán un valor de registro, esta a su vez define el orden de ingreso de dichos valores.
- VALUES ('Lista de Valores'): Aquí se registran los valores según las columnas especificadas en la tabla destino. Se pueden registrar valores separados por comas y especificar funciones de acuerdo a la capacidad declarada en la columna, por ejemplo, DEFAULT.

Caso 1: Ver algunos casos específicos de inserción de registros si se tiene la tabla Producto implementada tal como se muestra en la siguiente imagen:

Inserción de registros sin especificar las columnas de la tabla.

```
INSERT INTO PRODUCTO
          VALUES ('P0001','PYE DE MANZANA',20,500,'2015/05/14');
```

Caso 2: Inserción de registros especificando los nombres de las columnas de la tabla.

```
INSERT INTO PRODUCTO (id_producto, descripcion,precio,stock,fechavenc)
          VALUES ('P0001','PYE DE MANZANA',20,500,'2015/05/14');
```

Caso 3: Inserción de registros con especificación de valores por defecto.

```
INSERT INTO PRODUCTO
          VALUES ('P0001','PYE DE MANZANA',20,500,default);
```

Caso 4: Inserción de registros con una columna tipo fecha. Debe considerar que se registra el año, mes y día, respectivamente.

```
INSERT INTO PRODUCTO
          VALUES ('P0001','PYE DE MANZANA',20,500, '2015/05/14');
```

Caso 5: Inserción de registros con especificación del valor de autoincremento. Aquí es necesario especificar las columnas para registrar un valor. La columna que tiene el valor de autoincremento no será especificada. Suponiendo que el id_producto es declarado como autoincremento, entonces el registro del producto será de la siguiente manera:

```
INSERT INTO PRODUCTO (descripcion,precio,stock,fecha_venc)
          VALUES ('PYE DE MANZANA',20,500, '2015/05/14');
```

1.10.2 Sentencia UPDATE

A. Comentario

La sentencia UPDATE permite modificar o actualizar un conjunto de registros de una tabla. Estos cambios dependerán directamente de una condición sobre una o más columnas de la tabla. Se debe considerar los siguientes aspectos:

- Si la sentencia de actualización infringe una restricción, una regla o si el nuevo valor es de un tipo de datos incompatible al declarado en la tabla; se cancela la instrucción dando como resultado un mensaje error y no actualiza ningún registro de la tabla.
- La sentencia UPDATE es susceptible a errores aritméticos, que podría ser un error de desbordamiento o división por 0 durante la evaluación de la expresión, la actualización no se lleva a cabo mostrando en este caso un mensaje de error y corta la actualización desde el punto de error; a diferencia del primer punto este sí actualiza los registros hasta donde se ocasionó el error.

B. Sintaxis

```
UPDATE TABLA
    SET Columna = valor o expresión
    WHERE  Condición;
```

- TABLA: Aquí se especifica el nombre de la tabla que necesita ser actualizada.
- SET: Aquí se especifica la o las columnas que se actualizarán en una determinada tabla.
- Columna: Aquí se especifica el nombre de la columna que desea actualizar, solo hay que tener en cuenta que no se pueden actualizar las columnas con restricciones de autoincremento.
- Valor o expresión: Puede asignarse un valor directamente pero se debe considerar el tipo de datos que soporta la columna a actualizar, también puede usarse una expresión pero debemos considerar que dicha expresión solo emita un resultado.
- WHERE: Especifica la o las condiciones que limitarán los valores que se actualizarán.

Caso 1: Ver algunos casos específicos de inserción de registros si se tiene la tabla **Producto** implementada, tal como se muestra en la siguiente imagen:

La empresa ha adquirido el doble de stock de todos los productos y se nos ha encomendado actualizarlo. Se sabe que inicialmente los productos presentan el siguiente stock:

ID_PRODUCTO	DESCRIPCION	PRECIO	STOCK	FECHA_VENC
P0001	PYE DE MANZANA	20.00	500	2015-05-14
P0002	TORTA DE CHOCOLATE	45.00	100	2015-07-04
P0003	TORTA TRES LECHES	30.00	40	2015-06-24
P0004	MOUSE DE MANZANA	35.00	70	2015-09-06
P0005	ARROZ CON LECHE-ENVASE ESPECIAL	13.00	40	2015-11-04
P0006	MAZAMORRA MORADA	1.50	70	2015-12-04
P0007	YOGUR ARABE	3.00	100	2015-05-04

```
UPDATE PRODUCTO
    SET STOCK=STOCK*2;
```

ID_PRODUCTO	DESCRIPCION	PRECIO	STOCK	FECHA_VENC
P0001	PYE DE MANZANA	20.00	1000	2015-05-14
P0002	TORTA DE CHOCOLATE	45.00	200	2015-07-04
P0003	TORTA TRES LECHES	30.00	80	2015-06-24
P0004	MOUSE DE MANZANA	35.00	140	2015-09-06
P0005	ARROZ CON LECHE-ENVASE ESPECIAL	13.00	80	2015-11-04
P0006	MAZAMORRA MORADA	1.50	140	2015-12-04
P0007	YOGUR ARABE	3.00	200	2015-05-04

Como se observa, el stock de todos los productos se ha duplicado gracias a la sentencia UPDATE. Se debe tener en cuenta que al no colocar la sentencia condicional WHERE, la instrucción es aplicada a todos los registros.

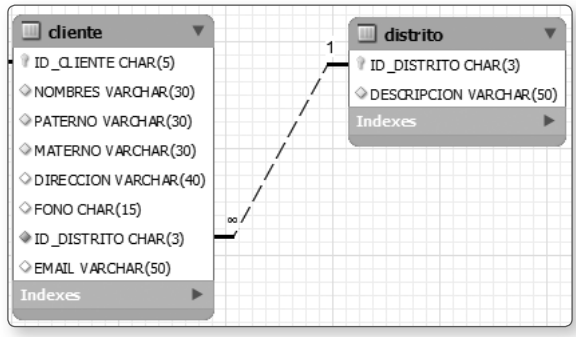
Caso 2: Modificar la fecha de vencimiento del producto cuyo código es 'P0002'.

```
UPDATE PRODUCTO
   SET FECHA_VENC='2015/10/05'
   WHERE ID_PRODUCTO='P0002';
```

Caso 3: Actualizar el precio del producto pye de manzana en un 20 % más de su precio actual.

```
UPDATE PRODUCTO
    SET PRECIO=PRECIO+PRECIO*0.2
    WHERE DESCRIPCION = 'PYE DE MANZANA';
```

Caso 4: Actualizar el código de distrito a L02 a todos los clientes cuyo distrito figure registrado el valor 'Cercado'.




```
UPDATE CLIENTE
    SET ID_DISTRITO='L02'
        WHERE ID_DISTRITO=(SELECT ID_DISTRITO
                                  FROM DISTRITO
                                  WHERE DESCRIPCION='Cercado');
```

1.10.3 Sentencia DELETE

A. Comentario

La sentencia DELETE permite eliminar todos los registros especificados en una determinada tabla. Se debe considerar los siguientes aspectos:

- La sentencia DELETE se puede bloquear cuando se ocasione un error de restricción FOREIGN KEY, es decir, se intenta eliminar registros que se encuentren asociados a otras tablas. Si se presentara este caso, el motor de base de datos envía un error al usuario y no elimina ningún registro del script.
- Cuando la sentencia DELETE determina un error aritmético en su proceso, el motor de base de datos cancela todo el proceso y envía un mensaje de error al usuario.
- También puede optar por la sentencia TRUNCATE TABLE cuando necesite eliminar todas las filas de una determinada tabla sin especificar la cláusula WHERE. En este caso, TRUNCATE usa menos recursos que DELETE, por lo tanto, es mucho más rápida la transacción.

B. Sintaxis

```
DELETE FROM nombre_tabla
    WHERE condición;
```

Caso 1: Eliminar todos los registros de la tabla Producto.

```
DELETE FROM producto;
```

Caso 2: Eliminar el producto con código 'p0002'.

```
DELETE FROM producto
    WHERE id_producto ='p0002';
```

Caso 3: Eliminar los productos cuya fecha de vencimiento sea inferior al día actual.

```
DELETE FROM PRODUCTO
    WHERE FECHA_VENC<curDate()
```

1.10.4 Sentencia TRUNCATE

A. Comentario

Permite eliminar todos los registros de una tabla. Se diferencia con la sentencia DELETE en el tiempo de respuesta.

B. Sintaxis

```
TRUNCATE nombre_tabla;
```

Caso: Eliminar todos los registros de la tabla Producto.

```
TRUNCATE producto;
```

1.11 Consulta básica con la sentencia SELECT

La sentencia SELECT en MySQL es la única que permite consultar información de forma que puede devolver un registro completo de datos o un solo valor. Se debe considerar que es la única función que permite obtener cualquier tipo de información desde las tablas de una base de datos.

1.11.1 Selección de datos sin restricciones

A. Comentario

Permite realizar una consulta a una determinada tabla sin condicionar la muestra del mismo.

B. Sintaxis

```
SELECT * FROM nombre_tabla;
```

El símbolo * representa a todos los registros, así como todas las columnas de una determinada tabla.

Caso 1: Listar todos los registros de la tabla Cliente.

```
SELECT * FROM CLIENTE;
```

Caso 2: Listar todos los clientes de la tabla Cliente usando un alias de la tabla.

```
SELECT C.* FROM CLIENTE C;
```

1.11.2 Selección de datos limitando las columnas de la tabla

A. Comentario

Este tipo de consulta permite especificar qué columnas se deben mostrar en la consulta.

B. Sintaxis

```
SELECT columnas FROM nombre_tabla;
```

Caso 1: De la tabla Producto, listar las columnas id_producto, descripción, stock.

```
SELECT id_producto, descripcion, stock FROM producto;
```

id_producto	descripcion	stock
P0001	PYE DE MANZANA	500
P0002	TORTA DE CHOCOLATE	100
P0003	TORTA TRES LECHES	40
P0004	MOUSE DE MANZANA	70
P0005	ARROZ CON LECHE-ENVASE ESPECIAL	40
P0006	MAZAMORRA MORADA	70

Caso 2: De la tabla Producto, listar las columnas id_producto, descripcion, stock, mostrando la siguiente cabecera de columnas:

Codigo del producto	Descripción del producto	Stock de producto
P0001	PYE DE MANZANA	500
P0002	TORTA DE CHOCOLATE	100
P0003	TORTA TRES LECHES	40
P0004	MOUSE DE MANZANA	70
P0005	ARROZ CON LECHE-ENVA...	40
P0006	MAZAMORRA MORADA	70
P0007	YOGUR ARABE	100

```
SELECT      id_producto as 'Código del producto',
            descripcion as 'Descripción del producto',
            stock as 'Stock de producto'
            FROM producto;
```

1.11.3 Restringir la muestra de datos repetidos

A. Comentario

Este tipo de consulta permite listar los registros de una tabla sin repetirse.

B. Sintaxis

```
SELECT DISTINCT nombre_columna FROM nombre_tabla;
```

Caso 1: Mostrar los códigos de los distritos registrados en la tabla Cliente sin repetir el código del mismo.

```
SELECT DISTINCT id_distrito
FROM cliente;
```

Caso 2: Mostrar los años obtenidos desde la fecha de vencimiento de los productos sin repetirse.

```
SELECT distinct YEAR(fecha_venc)
FROM producto;
```

1.11.4 Ordenar de forma ascendente o descendente los registros de una tabla

A. Comentario

Este tipo de consulta permite ordenar de forma ascendente o descendente los valores de una determinada columna de una tabla.

B. Sintaxis

```
SELECT * FROM nombre_tabla
    ORDER BY columna ASC|DESC;
```

Caso 1: Listar los productos ordenados en forma ascendente por la descripción del mismo.

```
SELECT * FROM producto
    ORDER BY descripcion ASC;
```

Caso 2: Listar los productos ordenados en forma descendente por el precio del mismo.

```
SELECT * FROM producto
    ORDER BY precio DESC;
```

Caso 3: Listar solo los años obtenidos desde la fecha de vencimiento de los productos. Es necesario tener en cuenta que no deban repetirse, además, deben mostrarse ordenados en forma descendente.

```
SELECT distinct year(fecha_venc) FROM producto
    ORDER BY year(fecha_venc) DESC;
```

1.11.5 Limitar los registros de la consulta

A. Comentario

Este tipo de consulta permitirá listar una cantidad de registros a partir de una consulta.

B. Sintaxis

```
SELECT * FROM nombre_tabla LIMIT numero_registros;
```

Caso 1: Listar los tres primeros productos registrados.

```
SELECT * FROM producto LIMIT 3;
```

Caso 2: Listar los tres productos más caros que se encuentran registrados.

```
SELECT * FROM producto ORDER BY precio DESC LIMIT 3;
```

Caso 3: Listar los tres primeros productos ordenados según su fecha de vencimiento de forma ascendente.

```
SELECT * FROM producto ORDER BY fecha_venc ASC LIMIT 3;
```

1.12 Combinación de tablas con INNER JOIN

A. Comentario

La combinación de tablas permite mostrar información cruzada entre dos o más tablas para diferentes procesos. Todo proviene de la llave foránea de una tabla, ya que es la que permitirá acceder a otra tabla y así combinarla.

B. Sintaxis

```
INNER JOIN <nombreTabla1>
    ON <nombreTabla2>
```

Caso 1: Listar la combinación de registros entre la tabla Cliente y Distrito.

```
SELECT * FROM CLIENTE C
       INNER JOIN DISTRITO D
       ON C.ID_DISTRITO = D.ID_DISTRITO;
```

ID_CLIENTE	NOMBRES	PATERNO	MATERNO	DIRECCION	FONO	ID_DISTRITO	EMAIL	ID_DISTRITO	DESCRIPCION
C0001	CARLOS	LOPEZ	HURTADO	Calle El Pino 346	4677352	L01	clopez@hotmail.com	L01	CERCADO
C0002	FRIDA	QUIROZ	DUARTE	Jr. Los Reyes 475	6734732	L06	fquiroz@peru.com	L06	CARABAYLLO
C0003	ALEJANDRO	TRAUCO	MANRIQUE	Jr.Huascar 956	4676732	L04	atruco@gmail.com	L04	BARRANCO
C0004	CARLA	BLANCO	RUIZ	Av. Arequipa 451	4867438	L03	cblanco@hotmail.com	L03	ATE
C0005	CORINA	MENDOZA	PEREZ	Av. Ingenierios 111	2548965	L09	cmendoza@hotmail.com	L09	CHORRILLOS
C0006	JORGE	RODAS	DIONICIO	Jr. Las Liras 456	4789658	L03	jrodas@gmail.com	L03	ATE
C0007	ORLANDO	CUEVAS	CABANILLAS	Calle La Encantada	5698532	L11	ocuevas@peru.com	L11	JESUS MARIA
C0008	ANTUANE	RODRIGUEZ	ALARCON	Av. Dorado 347	4589732	L05	arodriguez@hotmail.com	L05	BREÑA
C0009	MANUEL	SUAREZ	FERNANDEZ	Jr. Los Robles 854	4576738	L19	msuarez@gmail.com	L19	PACHACAMAC
C0010	JUAN	GUTIERREZ	DIAZ	Calle Girasoles 456	4897421	L14	jgutierrez@hotmail.com	L14	LINCE
C0011	CARLOS	COLAN	BARDALES	Av. Los Heroes 895	3698574	L19	ccolan@peru.com	L19	PACHACAMAC
C0012	MARTIN	CARRILLO	SALAS	Calle Los Huertos 84	8965952	L08	mcarrillo@gmail.com	L08	CHACLACAYO
C0013	JOSE	LAZARTE	LUJAN	Jr. Agapito 452	1258965	L17	jlazarte@gmail.com	L17	MAGDALENA
C0014	VIDAL	ZORRILLA	RODRIGUEZ	Av. Heroes del cene	1547894	L20	vzorrilla@hotmail.com	L20	PUCUSANA
C0015	GUILLERMO	RAMOS	FLORES	Jr. Las Almendras 21	4587964	L03	gramos@gmail.com	L03	ATE

Como se observa, el resultado muestra todas las columnas de la tabla Cliente de la misma forma que de la tabla Distrito, pero en dicha combinación se muestran todos los registros de clientes; mientras que, los registros de los distritos se limitan a mostrar aquellos que combinan con la tabla Cliente, es decir, no se listarán los distritos que no tengan asociación con los clientes.

Caso 2: Listar la combinación de registros entre la tabla Cliente y Distrito pero con la siguiente especificación de columnas: código del cliente, nombres y apellidos, descripción del distrito, correo electrónico del cliente; tal como se muestra en la siguiente imagen:

CODIGO	CLIENTE	DISTRITO	CORREO_ELECTRONICO
C0001	CARLOS LOPEZ HURTADO	CERCADO	clopez@hotmail.com
C0002	FRIDA QUIROZ DUARTE	CARABAYLLO	fquiroz@peru.com
C0003	ALEJANDRO TRAUCO MANRIQUE	BARRANCO	atruco@gmail.com
C0004	CARLA BLANCO RUIZ	ATE	cblanco@hotmail.com
C0005	CORINA MENDOZA PEREZ	CHORRILLOS	cmendoza@hotmail.com
C0006	JORGE RODAS DIONICIO	ATE	jrodas@gmail.com
C0007	ORLANDO CUEVAS CABANILLAS	JESUS MARIA	ocuevas@peru.com
C0008	ANTUANE RODRIGUEZ ALARCON	BREÑA	arodriguez@hotmail.com
C0009	MANUEL SUAREZ FERNANDEZ	PACHACAMAC	msuarez@gmail.com
C0010	JUAN GUTIERREZ DIAZ	LINCE	jgutierrez@hotmail.com
C0011	CARLOS COLAN BARDALES	PACHACAMAC	ccolan@peru.com
C0012	MARTIN CARRILLO SALAS	CHACLACAYO	mcarrillo@gmail.com
C0013	JOSE LAZARTE LUJAN	MAGDALENA	jlazarte@gmail.com
C0014	VIDAL ZORRILLA RODRIGUEZ	PUCUSANA	vzorrilla@hotmail.com
C0015	GUILLERMO RAMOS FLORES	ATE	gramos@gmail.com

```
SELECT C.ID_CLIENTE AS CODIGO,
       CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
       D.DESCRIPCION AS DISTRITO,
       C.EMAIL AS CORREO_ELECTRONICO
       FROM CLIENTE C
       INNER JOIN DISTRITO D
       ON C.ID_DISTRITO = D.ID_DISTRITO;
```

1.12.1 Combinación de tablas con LEFT JOIN

A. Comentario

La combinación con LEFT JOIN permite mostrar información de ambas tablas, haciendo que la tabla de la izquierda muestre todos sus registros; esto forzará a la tabla de la derecha a mostrar NULL en aquellos valores que no se combinen.

B. Sintaxis

```
SELECT * FROM <Tabla1>
    LEFT JOIN <Tabla2>
    ON Tabla1.columna = Tabla2.columna;
```

Caso 1: Listar la combinación de registros entre las tablas Cliente y Distrito usando combinación Left Join.

```
SELECT * FROM DISTRITO D
    LEFT JOIN CLIENTE C
    ON D.ID_DISTRITO=C.ID_DISTRITO;
```

ID_DISTRITO	DESCRIPCION	ID_CLIENTE	NOMBRES	PATERNO	MATERNO	DIRECCION	FONO	ID_DISTRITO	EMAIL
L01	CERCADO	C0001	CARLOS	LOPEZ	HURTADO	Calle El Pino 346	4677352	L01	clopez@hotmail.com
L02	ANCON	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
L03	ATE	C0004	CARLA	BLANCO	RUIZ	Av. Arequipa 451	4867438	L03	cblanco@hotmail.com
L03	ATE	C0006	JORGE	RODAS	DIONICIO	Jr. Las Liras 456	4789658	L03	jrodas@gmail.com
L03	ATE	C0015	GUILLERMO	RAMOS	FLORES	Jr. Las Almendras 211	4587964	L03	gramos@gmail.com
L04	BARRANCO	C0003	ALEJANDRO	TRAUCO	MANRIQUE	Jr.Huascar 956	4676732	L04	atruco@gmail.com
L05	BREÑA	C0008	ANTUANE	RODRIGUEZ	ALARCON	Av. Dorado 347	4589732	L05	arodriguez@hotmail.com
L06	CARABAYLLO	C0002	FRIDA	QUIROZ	DUARTE	Jr. Los Reyes 475	6734732	L06	fquiroz@peru.com
L07	COMAS	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
L08	CHACLACAYO	C0012	MARTIN	CARRILLO	SALAS	Calle Los Huertos 844	8965952	L08	mcarrillo@gmail.com
L09	CHORRILLOS	C0005	CORINA	MENDOZA	PEREZ	Av. Ingenierios 111	2548965	L09	cmendoza@hotmail.com
L10	EL AGUSTINO	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
L11	JESUS MARIA	C0007	ORLANDO	CUEVAS	CABANILLAS	Calle La Encantada 425	5698532	L11	ocuevas@peru.com
L12	LA MOLINA	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
L13	LA VICTORIA	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Al especificar el carácter asterisco, en la consulta se indica al servidor que debe mostrar todas las columnas y registros a ambas tablas. Por ser una combinación externa del lado izquierdo, la tabla Distrito muestra todos sus registros; aquí es donde la tabla Cliente se combina y muestra NULL en aquellas columnas que no tiene información; por ejemplo, el distrito ANCON no registra cliente, por lo tanto, en la información del mismo se muestra NULL en todas sus columnas.

Caso 2: Listar la descripción y el código de los distritos que no tienen registro en la tabla Cliente.

```
SELECT D.* FROM DISTRITO D
     LEFT JOIN CLIENTE C
     ON D.ID_DISTRITO=C.ID_DISTRITO
     WHERE C.ID_CLIENTE IS NULL;
```

Caso 3: Listar aquellos productos que aún no se registran en un detalle de boleta.

```
SELECT P.* FROM PRODUCTO P
     LEFT JOIN DETALLEBOLETA D
     ON P.ID_PRODUCTO = D.ID_PRODUCTO
     WHERE D.ID_PRODUCTO IS NULL;
```

1.12.2 Combinación de tablas con RIGHT JOIN

A. Comentario

La combinación con RIGHT JOIN muestra información de ambas tablas, y permite que la tabla de la derecha muestre todos sus registros, lo cual forzará a la tabla de la derecha a mostrar NULL en aquellos valores que no se combinen.

B. Sintaxis

```
SELECT * FROM <Tabla1>
    RIGTH JOIN <Tabla2>
    ON Tabla1.columna =  Tabla2.columna;
```

Caso 1: Listar la combinación de registros entre las tablas Cliente y Distrito usando la combinación Right Join.

```
SELECT * FROM CLIENTE C
     RIGHT JOIN DISTRITO D
     ON C.ID_DISTRITO=D.ID_DISTRITO;
```

ID_CLIENTE	NOMBRES	PATERNO	MATERNO	DIRECCION	FONO	ID_DISTRITO	EMAIL	ID_DISTRITO	DESCRIPCION
C0001	CARLOS	LOPEZ	HURTADO	Calle El Pino 346	4677352	L01	clopez@hotmail.com	L01	CERCADO
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	L02	ANCON
C0004	CARLA	BLANCO	RUIZ	Av. Arequipa 451	4867438	L03	cblanco@hotmail.com	L03	ATE
C0006	JORGE	RODAS	DIONICIO	Jr. Las Liras 456	4789658	L03	jrodas@gmail.com	L03	ATE
C0015	GUILLERMO	RAMOS	FLORES	Jr. Las Almendras 211	4587964	L03	gramos@gmail.com	L03	ATE
C0003	ALEJANDRO	TRAUCO	MANRIQUE	Jr.Huascar 956	4676732	L04	atruco@gmail.com	L04	BARRANCO
C0008	ANTUANE	RODRIGUEZ	ALARCON	Av. Dorado 347	4589732	L05	arodriguez@hotmail.com	L05	BREÑA
C0002	FRIDA	QUIROZ	DUARTE	Jr. Los Reyes 475	6734732	L06	fquiroz@peru.com	L06	CARABAYLLO
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	L07	COMAS
C0012	MARTIN	CARRILLO	SALAS	Calle Los Huertos 844	8965952	L08	mcarrillo@gmail.com	L08	CHACLACAYO
C0005	CORINA	MENDOZA	PEREZ	Av. Ingenierios 111	2548965	L09	cmendoza@hotmail.com	L09	CHORRILLOS

Al especificar el carácter asterisco, en la consulta se indica al servidor que debe mostrar todas las columnas y registros a ambas tablas. Por ser una combinación externa del lado derecho la tabla Distrito, muestra todos sus registros. Aquí es donde la tabla Cliente que se encuentra a la izquierda se combina y muestra NULL en aquellas columnas que no tiene información; por ejemplo, el distrito ANCON no registra cliente, por lo tanto, en la información del mismo se muestra NULL en todas sus columnas.

Caso 2: Listar la descripción y el código de los distritos que no tienen registro en la tabla Cliente usando Rigth Join.

```
SELECT D.* FROM CLIENTE C
     RIGHT JOIN DISTRITO D
     ON D.ID_DISTRITO=C.ID_DISTRITO
     WHERE C.ID_CLIENTE IS NULL;
```

Caso 3: Listar aquellos productos que aún no se registran en un detalle de boleta usando RIGTH JOIN.

```
SELECT P.* FROM DETALLEBOLETA D
     RIGHT JOIN PRODUCTO P
     ON P.ID_PRODUCTO = D.ID_PRODUCTO
     WHERE D.ID_PRODUCTO IS NULL;
```

1.13 Recuperación de datos agrupados

La idea de recuperar datos agrupados nace de la necesidad de mostrar informes o reportes a partir de dichas consultas, por ello, es necesario implementar correctamente los datos agrupados, para lo cual usaremos la cláusula GROUP BY.

1.13.1 Cláusula GROUP BY

A. Comentario

La cláusula GROUP BY agrupa un conjunto de registros de acuerdo a los valores de una o más columnas de una tabla. Para una mejor implementación de las consultas agrupadas, se puede usar las funciones agregadas como SUM, COUNT, MAX, MIN y AVG.

B. Sintaxis

```
SELECT columna_a_agrupar,funcion_agregada
     FROM tabla
     GROUP BY columna_a_agrupar;
```

- Columna_a_agrupar: Aquí se especifica el nombre de la columna a la que se desea crear una agrupación. Es necesario considerar que si hay más de una columna, estas deberán estar especificadas también en la cláusula GROUP BY.
- Función_agregada: Es una función estándar de MySQL como SUM, COUNT, MAX, MIN y AGV.

Caso 1: Listar el total de clientes agrupados por el código del distrito, tal como se muestra en la siguiente imagen:

ID_DISTRITO	COUNT(*)
L01	1
L03	3
L04	1
L05	1
L06	1
L08	1
L09	1
L11	1
L14	1
L17	1
L19	2
L20	1

```
SELECT ID_DISTRITO,COUNT(*)
    FROM CLIENTE
    GROUP BY ID_DISTRITO;
```

1.13.2 Función agregada COUNT

A. Comentario

La función COUNT permite contar un determinado número de registros según el criterio especificado por el usuario. Cuando no se especifica una condición, la función devuelve como respuesta el total de registros contenidos en la tabla.

B. Sintaxis

```
COUNT(*) o COUNT(nombre_columna);
```

Caso 1: Determinar el total de clientes registrados, tal como se muestra en la siguiente imagen:

TOTAL DE CLIENTES
15

```
SELECT COUNT(*) AS 'TOTAL DE CLIENTES'
    FROM CLIENTE;
```

Caso 2: Determinar el total de clientes registrados por cada distrito, tal como se muestra en la siguiente imagen:

DISTRITO	TOTAL DE CLIENTES
ATE	3
BARRANCO	1
BREÑA	1
CARABAYLLO	1
CERCADO	1
CHACLACAYO	1
CHORRILLOS	1
JESUS MARIA	1
LINCE	1
MAGDALENA	1
PACHACAMAC	2
PUCUSANA	1

```
SELECT D.DESCRIPCION AS 'DISTRITO', COUNT(*) AS 'TOTAL DE CLIENTES'
    FROM CLIENTE C
    INNER JOIN DISTRITO D
    ON C.ID_DISTRITO = D.ID_DISTRITO
    GROUP BY D.DESCRIPCION;
```

Caso 3: Determinar el total de boletas asociadas a los clientes en las cuales se debe mostrar el nombre completo del cliente, tal como se muestra en la siguiente imagen:

CLIENTE	TOTAL DE BOLETAS
CARLOS COLAN BARDALES	2
CARLOS LOPEZ HURTADO	4
CORINA MENDOZA PEREZ	3
FRIDA QUIROZ DUARTE	4
JORGE RODAS DIONICIO	7
MARTIN CARRILLO SALAS	1
VIDAL ZORRILLA RODRIGUEZ	1

```
SELECT CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
       COUNT(*) AS 'TOTAL DE BOLETAS'
       FROM BOLETA B
       INNER JOIN CLIENTE C
       ON B.ID_CLIENTE = C.ID_CLIENTE
       GROUP BY CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO);
```

La función CONCAT permite concatenar o unir dos o más columnas, para este caso se debe unir el nombre, apellido paterno y materno con un espacio entre ellos.

1.13.3 Función agregada SUM

A. Comentario

Permite devolver la suma de todos los valores contenidos en una columna de valores numéricos.

B. Sintaxis

```
SUM(Columna o Expresión)
```

Caso 1: Determinar el número total de productos que se encuentra en el área de almacén.

```
SELECT SUM(PRODUCTO.STOCK) AS 'TOTAL DE PRODUCTOS'
    FROM PRODUCTO;
```

Caso 2: Determinar el monto subtotal por cada boleta, este se dará en base al precio del producto y la cantidad registrada en el detalle de boleta, tal como se muestra en la siguiente imagen:

NUM_BOLETA	SUBTOTAL
1	530.00
2	1030.00
3	247.00
4	905.00
5	805.00
6	570.00
7	495.00
8	200.00
9	143.00
10	24.00
11	690.00

```
SELECT B.NUM_BOLETA, SUM(D.CANTIDAD*P.PRECIO) AS 'SUBTOTAL'
    FROM BOLETA B
    INNER JOIN DETALLEBOLETA D ON B.NUM_BOLETA=D.NUM_BOLETA
    INNER JOIN PRODUCTO P ON P.ID_PRODUCTO = D.ID_PRODUCTO
    GROUP BY B.NUM_BOLETA;
```

Caso 3: Determinar el monto total vendido por años, es decir, se debe listar los años registrados en la boleta y el subtotal por cada año, tal como se muestra en la siguiente imagen:

AÑO	SUBTOTAL
2015	10247.00

```
SELECT      YEAR(B.FECHA) AS 'AÑO',
            SUM(D.CANTIDAD*P.PRECIO) AS 'SUBTOTAL'
            FROM BOLETA B
            INNER JOIN DETALLEBOLETA D ON B.NUM_BOLETA=D.NUM_BOLETA
            INNER JOIN PRODUCTO P ON P.ID_PRODUCTO = D.ID_PRODUCTO
            GROUP BY YEAR(B.FECHA);
```

1.13.4 Función agregada MAX

A. Comentario

Esta función permite determinar el valor máximo encontrado en una columna de valores numéricos.

B. Sintaxis

```
MAX (Columna o Expresion)
```

- Columna: Se puede especificar el nombre de una columna de la tabla al buscar el mayor valor.
- Expresion: Representa a una función SQL o a un juego de operadores aritméticos.
- MAX solo puede ser usado con columnas o expresiones que tengan como tipo un entero, en caso contrario, genera un error proveniente desde el motor de base de datos.

Caso 1: Determinar el precio más alto registrado en la tabla Producto.

```
SELECT MAX(PRECIO) AS 'PRECIO ALTO'
    FROM PRODUCTO;
```

Caso 2: Determinar el monto subtotal más alto que registre un determinado cliente, tal como se muestra en la siguiente imagen:

CLIENTE	SUBTOTAL MAS ALTO
CARLOS COLAN BARDALES	400.00
CARLOS LOPEZ HURTADO	675.00
CORINA MENDOZA PEREZ	380.00
FRIDA QUIROZ DUARTE	900.00
JORGE RODAS DIONICIO	570.00
MARTIN CARRILLO SALAS	420.00
VIDAL ZORRILLA RODRIGUEZ	60.00

```
SELECT CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
    MAX(D.CANTIDAD*P.PRECIO) AS 'SUBTOTAL MAS ALTO'
    FROM BOLETA B
    INNER JOIN CLIENTE C ON B.ID_CLIENTE = C.ID_CLIENTE
    INNER JOIN DETALLEBOLETA D ON B.NUM_BOLETA = D.NUM_BOLETA
    INNER JOIN PRODUCTO P ON D.ID_PRODUCTO = P.ID_PRODUCTO
    GROUP BY CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO);
```

1.13.5 Función agregada MIN

A. Comentario

Esta función permite determinar el valor mínimo encontrado en una columna de valores numéricos o una expresión.

B. Sintaxis

```
MIN (Columna o Expresion)
```

- Columna: Se puede especificar el nombre de una columna de la tabla al buscar el menor valor.
- Expresion: Representa a una función SQL o a un juego de operadores aritméticos.
- MIN solo puede ser usado con columnas o expresiones que tengan como tipo un entero; en caso contrario, genera un error proveniente desde el motor de base de datos.

Caso 1: Listar el precio del producto más bajo.

```
SELECT MIN(PRECIO) AS 'PRECIO BAJO'
    FROM PRODUCTO;
```

Caso 2: Determinar el monto subtotal más bajo que registre un determinado cliente, tal como se muestra en la siguiente imagen:

CLIENTE	SUBTOTAL MAS BAJO
CARLOS COLAN BARDALES	24.00
CARLOS LOPEZ HURTADO	130.00
CORINA MENDOZA PEREZ	30.00
FRIDA QUIROZ DUARTE	130.00
JORGE RODAS DIONICIO	30.00
MARTIN CARRILLO SALAS	273.00
VIDAL ZORRILLA RODRIGUEZ	60.00

```
SELECT CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
       MIN(D.CANTIDAD*P.PRECIO) AS 'SUBTOTAL MAS ALTO'
       FROM BOLETA B
       INNER JOIN CLIENTE C ON B.ID_CLIENTE = C.ID_CLIENTE
       INNER JOIN DETALLEBOLETA D ON B.NUM_BOLETA = D.NUM_BOLETA
       INNER JOIN PRODUCTO P ON D.ID_PRODUCTO = P.ID_PRODUCTO
       GROUP BY CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO);
```

1.13.6 Función agregada AVG

A. Comentario

Devuelve el promedio de los valores de un determinado grupo. Se debe considerar que dicha columna debe ser numérica.

B. Sintaxis

```
AVG (Columna o Expresion)
```

- Columna: Se puede especificar el nombre de una columna para determinar el promedio.
- Expresion: Representa a una función SQL o a un juego de operadores aritméticos. No se permite las funciones de agregado ni subconsultas.

Caso 1: Determinar el promedio de precios de todos los productos.

```
SELECT AVG(PRECIO) AS 'PROMEDIO DE PRECIOS'
    FROM PRODUCTO;
```

1.13.7 Cláusula HAVING

A. Comentario

Permite condicionar los resultados a partir de una agrupación, es decir, evaluará qué registros debe mostrar después de la agrupación de valores. Normalmente, HAVING es usado después de la cláusula GROUP BY.

B. Sintaxis

```
HAVING <condición>
```

Caso 1: Determinar el monto total vendido en el 2015, tal como se muestra en la siguiente imagen:

AÑO	SUBTOTAL
2015	10247.00

```
SELECT      YEAR(B.FECHA) AS 'AÑO',
            SUM(D.CANTIDAD*P.PRECIO) AS 'SUBTOTAL'
            FROM BOLETA B
            INNER JOIN DETALLEBOLETA D ON B.NUM_BOLETA=D.NUM_BOLETA
            INNER JOIN PRODUCTO P ON P.ID_PRODUCTO = D.ID_PRODUCTO
            GROUP BY YEAR(B.FECHA)
            HAVING AÑO = 2015;
```

CAP.

2

Integración MySQL con PHP

2.1 Introducción

PHP es el lenguaje de programación que soporta muchos gestores de base de datos, pero el más usado es MySQL; por ese motivo, se ha implementado un capítulo completo de las principales funciones de MySQL que servirán para el desarrollo de los siguientes capítulos.

No se pretende crear una dependencia directa con MySQL sino más bien demostrar la facilidad de integración entre MySQL y PHP, ya que está catalogado como uso popular. En los posteriores capítulos se hará referencia al acceso que PHP puede tener con PostGre y SQL Server.

2.2 Conectar MySQL a PHP

PHP cuenta con las funciones necesarias para conectarse a distintos motores de base de datos y ahora conoceremos las principales para integrarlo a MySQL. Es necesario, hasta aquí, tener experiencia en creación de base de datos, tablas, relaciones y manipulación de los datos en MySQL; en caso contrario, se recomienda leer el primer capítulo de este libro.

2.2.1 Función de conexión al servidor MySQL

A. Comentario

Conectarse a un servidor de base de datos conlleva tener ciertos requisitos previos como el nombre del servidor, el nombre del usuario y su respectiva contraseña; sin embargo, como se indicó en el apartado anterior, si tiene conocimientos previos de MySQL, entonces la sintaxis mostrada para la conexión al servidor será sencilla.

B. Sintaxis

```
mysql_connect('SERVIDOR','USUARIO','CLAVE');
```

- SERVIDOR: Es el nombre del servidor. Si tiene como servidor a WAMP Server o XAMPP, el nombre del servidor es «localhost».
- USUARIO: Es el nombre del usuario de la base de datos. En este caso es «root».
- CLAVE: Al iniciar MySQL no asigne alguna clave, por lo tanto, este campo quedará vacío a menos que lo hubiera modificado.

Caso 1: Conectarse al servidor MySQL de forma básica.

Archivo: **conexion1.php**

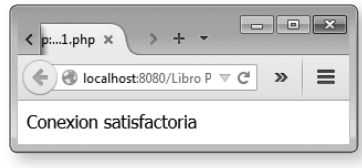
```
<!DOCTYPE html>
<html>
   <head>
       <meta charset="UTF-8">
       <title></title>
       <link href="estilo.css" rel="stylesheet">
   </head>
    <body>
        <?php
```

```
            $servidor='localhost';
            $usuario='root';
            $clave='';

            mysql_connect($servidor,$usuario,$clave);
        ?>
    </body>
</html>
```

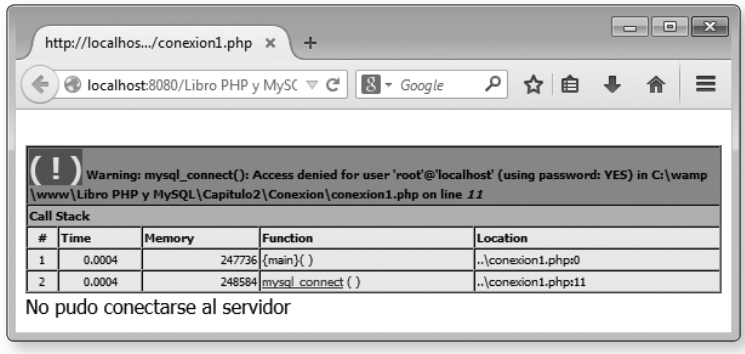
Se cuenta con el archivo **estilo.css** incorporado en la misma carpeta del archivo **conexion1.php**.

```
body {
    font-family: tahoma;
    font-size: 14;
}
```

Caso 2: Conectarse al servidor MySQL mostrando el mensaje «Conexión satisfactoria» si todo es correcto; en caso contrario, mostrará el mensaje «No pudo conectarse al servidor».



En caso hubiera algún error en el servidor, usuario o clave, se mostrará la siguiente imagen:



Archivo: **conexion2.php**

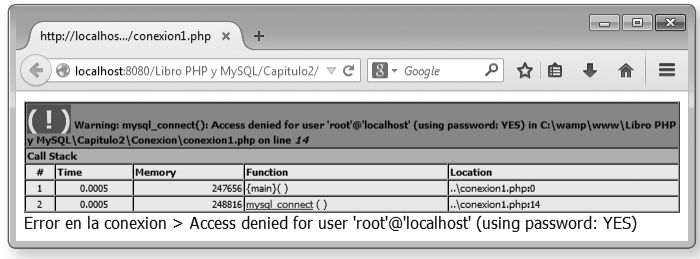
```
<?php
    $cn=mysql_connect('localhost','root','');
    if (!$cn)
       echo "No pudo conectarse al servidor";
       else
       echo "Conexión satisfactoria";
?>
```

En esta conexión agregue directamente el nombre del servidor, del usuario y la clave a la función **mysql connect**.

El resultado de conectarse a la base de datos es un valor booleano que en este caso ha sido almacenado en la variable **$cn** que, a su vez, es evaluada en la condicional **if** para determinar si todos los datos han sido correctos. Considere la siguiente tabla:

- if ($cn) > devuelve *true*: conexión correcta
- if (!$cn) > devuelve *false*: conexión incorrecta

Caso 3: Conectarse al servidor emitiendo el mensaje de «Error de conexión», además del comentario interno de PHP sobre el error, solo si hay un error en la conexión, tal como se muestra en la siguiente imagen:



Archivo: **conexion3.php**

```
<?php
  $servidor='localhost';
  $usuario='root';
  $clave='xxxx';

  $cn=mysql_connect($servidor,$usuario,$clave) or
              die('Error en la conexión > '.mysql_error());
?>
```

La función **die** es la encargada de mostrar un mensaje, además de detener la ejecución de las demás sentencias PHP.

La función **mysql_error()** permite mostrar un mensaje de error interno entendido por PHP al momento de conectarse al servidor.

2.2.2 Función de cierre del servidor

A. Comentario

Toda conexión correcta a un servidor debe ser cerrada para un próximo uso. Mantenerla abierta conlleva a posibles errores cuando vuelva a conectarse al servidor. Esto es similar a intentar abrir una puerta cuando se encuentra abierta, ya que lo lógico sería que primero se cierre dicha puerta y después se abra.

B. Sintaxis

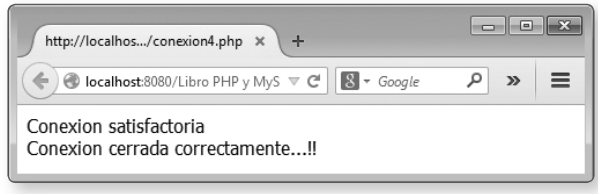
```
mysql_close($variable_de_conexión_al_servidor);
```

Caso 1: Conectarse al servidor mostrando el mensaje «Conexión satisfactoria» si así lo fuera; en caso contrario, mostrará «No pudo conectarse al servidor» y, además, el mensaje de cerrar la conexión.

Archivo: **conexion4.php**

```
<?php
    $cn=mysql_connect('localhost','root','');
    if (!$cn)
       echo "No pudo conectarse al servidor";
       else
       echo "Conexión satisfactoria";
    mysql_close($cn);
?>
```

Caso 2: Conectarse al servidor mostrando el mensaje «Conexión satisfactoria» si así lo fuera; en caso contrario, mostrará el mensaje «No pudo conectarse al servidor» y, además, el mensaje «Conexión cerrada correctamente», tal como se muestra en la siguiente imagen:



Archivo: **conexion4.php**

```
<?php
    $cn=mysql_connect('localhost','root','');
    if (!$cn)
       echo 'No pudo conectarse al servidor';
       else
       echo 'Conexión satisfactoria';

    $ccn=mysql_close($cn);
    if ($ccn)
        echo '<br>Conexión cerrada correctamente...!!';
?>
```

2.2.3 Función de selección de base de datos desde el servidor

A. Comentario

La función **mysql_select_db()** selecciona una base de datos MySQL existente. Para ello previamente debe haberse conectado con el servidor MySQL a través de **mysql_connect**.

Devolverá TRUE si todo se llevó a cabo correctamente o FALSE si se produjo algún fallo.

B. Sintaxis

```
mysql_select_db(nombre_base_de_datos,variable_de_conexion);
```

- nombre_base_de_datos: Es el nombre de la base de datos implementada en el servidor actual.
- variable_de_conexion: Es la variable que usa la función **mysql_connect** previamente validada.

Caso 1: Seleccionar la base de datos Ventas2015 de forma básica.

Archivo: **selecciona1.php**

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
        <link href="estilo.css" rel="stylesheet">
    </head>
 <?php
    /* Conectándose al servidor */
    $cn=mysql_connect('localhost','root','');

    /* Seleccionando la base de datos Ventas2015 */
    mysql_select_db('ventas2015',$cn);

    /* Cerrando la conexión */
    mysql_close($cn);
 ?>
    </body>
</html>
```

Caso 2: Seleccionar la base de datos Ventas2015 emitiendo el mensaje «Base de datos seleccionada correctamente...!!!», en caso contrario, mostrará el mensaje «La base de datos NO existe».

Archivo: **selecciona2.php**

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
        <link href="estilo.css" rel="stylesheet">
    </head>
 <?php
    /* Conectándose al servidor */
    $cn=mysql_connect('localhost','root','') or
                            die('Error de conexión '.  mysql_error());

    /* Seleccionando la base de datos Ventas2015 */
    $db=mysql_select_db('ventas2015',$cn);

    /* Validando la existencia de la base de datos */
    if(!$db)
        echo 'La base de datos No existe';
    else
       echo 'Base de datos seleccionada correctamente..!!!';

    /* Cerrando la conexión */
    mysql_close($cn);
 ?>
    </body>
</html>
```

2.2.4 Función que ejecuta una sentencia de consulta MySQL en PHP

A. Comentario

Una vez realizada la conexión y seleccionada una base de datos desde el servidor, puede recuperar información desde las tablas de la base de datos activa; para ello, use la función **mysql_query** que permite ejecutar una sentencia de consulta al servidor.

B. Sintaxis

```
$rs=mysql_query(sentencia_mysql,variable_de_conexion);
```

- La variable **$rs** almacena el valor booleano de la consulta, es decir, TRUE si la consulta es correcta o FALSE si está errada.
- La función retornará un conjunto de registros solo si la **sentencia_mysql** es SELECT, en caso ocurra un error en dicha sentencia, esta devolverá FALSE como resultado.
- La función retornará TRUE solo si la **sentencia_mysql** es INSERT, UPDATE o DELETE, en caso encuentre un error en dicha sentencia, esta devolverá FALSE como resultado.
- Se debe tener en cuenta que la función **mysql_query** envía la consulta al servidor MySQL, pero no muestra los resultados en la página HTML.

Caso 1: Ejecutar la sentencia MySQL que permite mostrar todos los registros de la tabla Cliente.

Archivo: **consulta1.php**

```
<?php
    /* Conectándose al servidor */
    $cn=mysql_connect('localhost','root','');

    /* Seleccionando la base de datos Ventas2015 */
    mysql_select_db('ventas2015',$cn);

    /* Ejecutando la consulta */
    $rs = mysql_query('SELECT * FROM CLIENTE',$cn);

    /* Cerrando la conexión */
    mysql_close($cn);
 ?>
```

Caso 2: Ejecutar la sentencia MySQL que permite mostrar todos los registros de la tabla Cliente usando una variable para la sentencia SQL.

Archivo: **consulta2.php**

```
<?php
    /* Conectándose al servidor */
    $cn=mysql_connect('localhost','root','');

    /* Seleccionando la base de datos Ventas2015 */
    mysql_select_db('ventas2015',$cn);

    /* Ejecutando la consulta */
    $sql='SELECT * FROM CLIENTE';
    $rs = mysql_query($sql,$cn);
```

```
    /* Cerrando la conexión */
    mysql_close($cn);
?>
```

Caso 3: Ejecutar la sentencia MySQL que permite mostrar todos los registros de la tabla Cliente, además deberá mostrar el mensaje «Consulta implementada correctamente» si la sentencia de consulta es correcta, en caso contrario, emitirá el mensaje «Error en la implementación de la consulta».

Archivo: **consulta3.php**

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
            /* Conexión al servidor */
            $cn=mysql_connect('localhost','root','');

            /* Seleccionando la base de datos Ventas2015 */
            mysql_select_db('ventas2015',$cn);

            /* Consultando la tabla cliente */
            $rs=mysql_query('SELECT * FROM CLIENTE',$cn);

            /* Validando la sentencia de consulta */
            if ($rs)
                echo 'Consulta implementada correctamente';
            else
                echo 'Error en la implementación de la consulta';
        ?>
    </body>
</html>
```

2.2.5 Función que determine el total de registros devueltos por la función mysql_query usando la sentencia SELECT de MySQL

A. Comentario

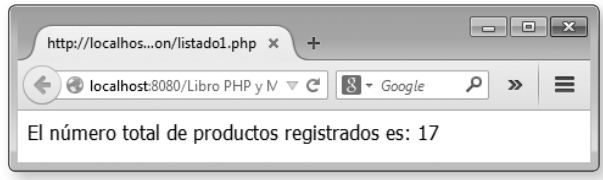
Será de mucha importancia determinar el número de registros encontrados por la función **mysql_query**, ya que desde aquí puede determinar exactamente cuántos registros vienen de la consulta o cuántos registros fueron afectados por la misma.

B. Sintaxis

```
$variable = mysql_num_rows(variable_de_consulta);
```

- $variable: Es el número total de registros resultantes de una consulta.
- variable_de_consulta: Es aquella variable que ejecuta la función **mysql_query**.

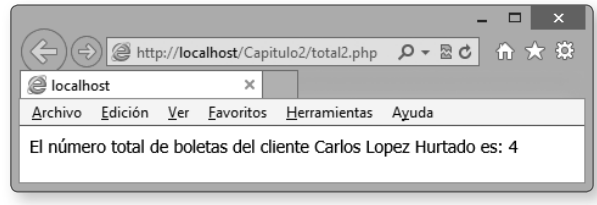
Caso 1: Determinar el número total de productos, tal como se muestra en la siguiente imagen:



Archivo: **total1.php**

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
        <link href="estilo.css" rel="stylesheet">
    </head>
    <body>
        <?php
            /* Conexión al servidor */
            $cn=mysql_connect('localhost','root','');

            /* Seleccionando la base de datos Ventas2015 */
            mysql_select_db('ventas2015',$cn);

            /* Consultando la tabla cliente */
            $rs=mysql_query('SELECT * FROM PRODUCTO',$cn);
            $n=  mysql_num_rows($rs);
            echo 'El número total de productos registrados es: '.$n;
        ?>
    </body>
</html>
```

Caso 2: Determinar el número total de boletas asociadas al cliente Carlos López Hurtado, tal como se muestra en la siguiente imagen:



Archivo: **total2.php**

```
<!DOCTYPE html>
<html>
 <head>
    <meta charset="UTF-8">
    <title>
</title>
    <link href="estilo.css" rel="stylesheet">
 </head>
 <body>
  <?php
```

```
    /* Conexión al servidor */
    $cn = mysql_connect('localhost','root','');

    /* Seleccionando la base de datos Ventas2015 */
    mysql_select_db('ventas2015',$cn);

    /* Consultando la tabla cliente */
    $cliente='Carlos López Hurtado';
    $rs = mysql_query("SELECT B.*
                       FROM BOLETA B
                       INNER JOIN CLIENTE C
                       ON B.ID_CLIENTE=C.ID_CLIENTE
        WHERE CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO)='$cliente'",$cn);

    /* Determinar el total de registros según la consulta */
    $n = mysql_num_rows($rs);
    echo 'El número total de boletas del cliente '.$cliente.' es: '.$n;
   ?>
  </body>
 </html>
```

2.2.6 Función que muestre los registros de una consulta mysql_query en PHP

A. Comentario

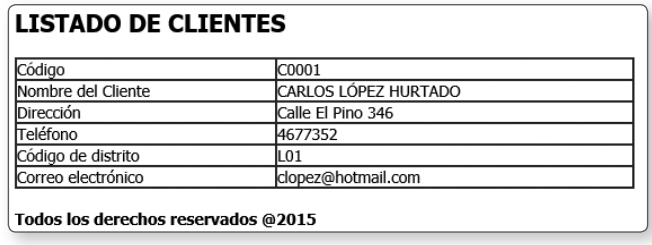
La ejecución de una consulta al servidor conlleva el manejo de registros de tamaño variable, es decir, podría devolver un juego de registros como también podría devolver un solo valor. En ambos casos, se necesita una función que se introduzca dentro de estos registros y los pueda devolver, de forma que puedan ser visibles por el usuario, ya sea en controles independientes del HTML o en una tabla de registros.

B. Sintaxis

```
mysql_result(variable_consulta,
             numero_registro,
             nombre_columna);
```

- variable_consulta: Es la referencia a la consulta ejecutada con la función **mysql_query()**.
- numero_registro: Es el número de registro desde el cual se desea obtener su información, tenga en cuenta que el primer registro se ubica en la posición 0 y el último registro en la posición n-1, donde n es el número total de registros obtenidos por la consulta.
- nombre_columna: Se obtiene de los nombres de la columna de la tabla consultada, tenga en cuenta que dichas columnas podrían ser tratadas como números enteros iniciando en 0 para la primera columna.

Caso 1: Listar el primer cliente registrado, tal como se muestra en la siguiente imagen:



LISTADO DE CLIENTES

Código	C0001
Nombre del Cliente	CARLOS LÓPEZ HURTADO
Dirección	Calle El Pino 346
Teléfono	4677352
Código de distrito	L01
Correo electrónico	clopez@hotmail.com

Todos los derechos reservados @2015

Archivo: **listado1.php**

```
<!DOCTYPE html>
<html>
  <head>
        <meta charset="UTF-8">
        <title></title>
        <link href="estilo.css" rel="stylesheet">
  </head>
  <body>
    <header>
       <h2>LISTADO DE CLIENTES</h2>
    </header>
    <section>
       <?php
         /* Conexión al servidor */
         $cn = mysql_connect('localhost','root','');

         /* Seleccionando la base de datos Ventas2015 */
         mysql_select_db('ventas2015',$cn);

         /* Consultando la tabla cliente */
         $rs = mysql_query('SELECT * FROM cliente',$cn);
       ?>
       <table border="1" width="550"
              cellspacing="0" cellpadding="0">
         <tr>
           <td>Codigo</td>
           <td><?php echo mysql_result($rs, 0,'id_cliente'); ?></td>
         </tr>
         <tr>
           <td>Nombre del Cliente</td>
           <td><?php echo mysql_result($rs, 0,'nombres').' '.
                          mysql_result($rs, 0,'paterno').' '.
                          mysql_result($rs, 0,'materno'); ?></td>
         </tr>
         <tr>
           <td>Dirección</td>
           <td><?php echo mysql_result($rs, 0,'direccion'); ?></td>
         </tr>
         <tr>
           <td>Teléfono</td>
           <td><?php echo mysql_result($rs, 0,'fono'); ?></td>
         </tr>
         <tr>
           <td>Codigo de distrito</td>
           <td><?php echo mysql_result($rs, 0,'id_distrito'); ?></td>
         </tr>
```

```
          <tr>
            <td>Correo electrónico</td>
            <td><?php echo mysql_result($rs, 0,'email'); ?></td>
          </tr>
        </table>
      </section>
      <footer>
        <h4>Todos los derechos reservados @2015</h4>
      </footer>
    </body>
</html>
```

Observación: Tenga en cuenta que el valor 0 hace referencia a la primera fila de registros obtenidos a partir de la consulta.

Caso 2: Implementar una aplicación web que permita listar los clientes así como también determinar el total de clientes, tal como se muestra en la siguiente imagen:

LISTADO DE CLIENTES

CÓDIGO	CLIENTE	DIRECCIÓN	TELÉFONO	DISTRITO	CORREO ELECTRÓNICO
C0001	CARLOS LOPEZ HURTADO	Calle El Pino 346	4677352	L01	clopez@hotmail.com
C0002	FRIDA QUIROZ DUARTE	Jr. Los Reyes 475	6734732	L06	fquiroz@peru.com
C0003	ALEJANDRO TRAUCO MANRIQUE	Jr.Huascar 956	4676732	L04	atruco@gmail.com
C0004	CARLA BLANCO RUIZ	Av. Arequipa 451	4867438	L03	cblanco@hotmail.com
C0005	CORINA MENDOZA PEREZ	Av. Ingenierios 111	2548965	L09	cmendoza@hotmail.com
C0006	JORGE RODAS DIONICIO	Jr. Las Liras 456	4789658	L03	jrodas@gmail.com
C0007	ORLANDO CUEVAS CABANILLAS	Calle La Encantada 425	5698532	L11	ocuevas@peru.com
C0008	ANTUANE RODRIGUEZ ALARCON	Av. Dorado 347	4589732	L05	arodriguez@hotmail.com
C0009	MANUEL SUAREZ FERNANDEZ	Jr. Los Robles 854	4576738	L19	msuarez@gmail.com
C0010	JUAN GUTIERREZ DIAZ	Calle Girasoles 456	4897421	L14	jgutierrez@hotmail.com
C0011	CARLOS COLAN BARDALES	Av. Los Heroes 895	3698574	L19	ccolan@peru.com
C0012	MARTIN CARRILLO SALAS	Calle Los Huertos 844	8965952	L08	mcarrillo@gmail.com
C0013	JOSE LAZARTE LUJAN	Jr. Agapito 452	1258965	L17	jlazarte@gmail.com
C0014	VIDAL ZORRILLA RODRIGUEZ	Av. Heroes del cenepa 635	1547894	L20	vzorrila@hotmail.com
C0015	GUILLERMO RAMOS FLORES	Jr. Las Almendras 211	4587964	L03	gramos@gmail.com
EL TOTAL DE CLIENTES ES: 15					

Todos los derechos reservados @2015

Archivo: **listado2.php**

```
<!DOCTYPE html>
<html>
 <head>
   <meta charset="UTF-8">
   <title></title>
   <link href="miEstilo.css" rel="stylesheet">
 </head>
 <body>
```

```
    <header>
       <img src="banner_clientes.png" width="940" height="210"/>
       <h2>LISTADO DE CLIENTES</h2>
    </header>
    <section>
      <?php
         /* Conexión al servidor */
         $cn = mysql_connect('localhost','root','');

         /* Seleccionando la base de datos Ventas2015 */
         mysql_select_db('ventas2015',$cn);

         /* Consultando la tabla cliente */
         $rs = mysql_query('SELECT * FROM cliente',$cn);

         /* Determinar el total de clientes */
         $n = mysql_num_rows($rs);
      ?>
      <table border="0" width="650" cellspacing="0" cellpadding="0">
        <tr>
          <td>CODIGO</td>
          <td>CLIENTE</td>
          <td>DIRECCION</td>
          <td>TELEFONO</td>
          <td>DISTRITO</td>
          <td>CORREO ELECTRONICO</td>
        </tr>
        <?php for($i=0;$i<$n;$i++){ ?>
        <tr>
          <td><?php echo mysql_result($rs, $i,'id_cliente'); ?></td>
          <td>
             <?php echo mysql_result($rs, $i,'nombres').' '.
                        mysql_result($rs, $i,'paterno').' '.
                        mysql_result($rs, $i,'materno');
             ?>
          </td>
             <td><?php echo mysql_result($rs, $i,'direccion'); ?></td>
             <td><?php echo mysql_result($rs, $i,'fono'); ?></td>
             <td>
                <?php echo mysql_result($rs, $i,'id_distrito'); ?>
             </td>
             <td><?php echo mysql_result($rs, $i,'email'); ?></td>
        </tr>
        <?php } ?>
        <tr>
             <td colspan="6" id="texto">
                <?php echo 'EL TOTAL DE CLIENTES ES: '.$n; ?>
             </td>
        </tr>
        </table>
        </section>
        <footer>
             <h4>Todos los derechos reservados @2015</h4>
        </footer>
     </body>
  </html>
```

Observación: Para que pueda mostrar todos los registros de clientes, primero debe determinar el total de clientes obtenidos desde la consulta, esto lo realiza con la función **mysql_num_rows**, luego lo estructura en la sentencia **for** para listar a todos los clientes, reemplazando el número 0 del caso anterior por la variable $i.

Archivo: **miEstilo.css**

```
body {
    font-family: tahoma;
    font-size: 12px;
}
table {
   width: 100%;
   border: 1px solid #000;
}
tr {
   background: #eee;
}
```

Caso 3: Implementar una aplicación web que permita listar el código, nombre completo de los clientes, la descripción del distrito y mostrar el total de clientes, tal como se muestra en la siguiente imagen:

LISTADO DE CLIENTES

CÓDIGO	CLIENTE	DISTRITO
C0001	CARLOS LOPEZ HURTADO	CERCADO
C0002	FRIDA QUIROZ DUARTE	CARABAYLLO
C0003	ALEJANDRO TRAUCO MANRIQUE	BARRANCO
C0004	CARLA BLANCO RUIZ	ATE
C0005	CORINA MENDOZA PEREZ	CHORRILLOS
C0006	JORGE RODAS DIONICIO	ATE
C0007	ORLANDO CUEVAS CABANILLAS	JESUS MARIA
C0008	ANTUANE RODRIGUEZ ALARCON	BREÑA
C0009	MANUEL SUAREZ FERNANDEZ	PACHACAMAC
C0010	JUAN GUTIERREZ DIAZ	LINCE
C0011	CARLOS COLAN BARDALES	PACHACAMAC
C0012	MARTIN CARRILLO SALAS	CHACLACAYO
C0013	JOSE LAZARTE LUJAN	MAGDALENA
C0014	VIDAL ZORRILLA RODRIGUEZ	PUCUSANA
C0015	GUILLERMO RAMOS FLORES	ATE

EL TOTAL DE CLIENTES ES: 15

Todos los derechos reservados @2015

Archivo: **listado3.php**

```
<!DOCTYPE html>
<html>
 <head>
   <meta http-equiv="Content-Type"
         content="text/html; charset=ISO-8859-1" />
   <title></title>
   <link href="miEstilo.css" rel="stylesheet">
 </head>
 <body>
   <header>
      <img src="banner_clientes.png" width="940" height="210"/>
      <h2>LISTADO DE CLIENTES</h2>
   </header>
   <section>
    <?php
      /* Conexión al servidor */
```

```
        $cn = mysql_connect('localhost','root','');

        /* Seleccionando la base de datos Ventas2015 */
        mysql_select_db('ventas2015',$cn);

        /* Consultando la tabla cliente */
        $rs = mysql_query("SELECT C.ID_CLIENTE,
                        CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO),
                        D.DESCRIPCION
                        FROM CLIENTE C INNER JOIN DISTRITO D
                        ON C.ID_DISTRITO=D.ID_DISTRITO",$cn);

          /* Determinar el total de clientes */
          $n = mysql_num_rows($rs);
      ?>
      <table border="0" width="650" cellspacing="0" cellpadding="0">
        <tr>
          <td id="fila">CODIGO</td>
          <td id="fila">CLIENTE</td>
          <td id="fila">DISTRITO</td>
        </tr>
        <?php for($i=0;$i<$n;$i++){ ?>
        <tr>
          <td><?php echo mysql_result($rs, $i,0);?></td>
          <td><?php echo mysql_result($rs, $i,1);?></td>
          <td><?php echo mysql_result($rs, $i,2);?></td>
         </tr>
         <?php } ?>
         <tr>
             <td colspan="6" id="texto">
                 <?php echo 'EL TOTAL DE CLIENTES ES: '.$n; ?>
             </td>
         </tr>
         </table>
         </section>
         <footer>
             <h4>Todos los derechos reservados @2015</h4>
         </footer>
    </body>
</html>
```

Observación: Se observa que la función **mysql_result** hace referencia a un valor numérico 0, 1, 2, respectivamente; esto se debe a la cantidad de columnas que emite la consulta; así tenemos `0` obtiene el código, 1 al nombre completo y 2 a la descripción del distrito de los clientes.

Archivo: **miEstilo.css**

```
body {
    font-family: tahoma;
    font-size: 12px;
}
table {
   width: 100%;
   border: 1px solid #000;
}
#fila {
   background: activecaption;
   font-size: 14px;
}
```

2.2.7 Casos desarrollados

Caso desarrollado 1 **Listado de clientes por distrito**

Implementar una aplicación web con PHP y MySQL que permita listar los clientes según el distrito seleccionado por el usuario, tal como se muestra en la siguiente imagen:

Imagen: ListadoClientesxDistrito.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **ClientesxDistrito**.
- Los nombres de los distritos deben encontrarse precargados en el control cuadro combinado; se debe tener en cuenta que el código del distrito representa su valor y, finalmente, considere que solo se deben mostrar los distritos en los cuales haya por lo menos un cliente registrado.
- Al seleccionar un distrito desde el control cuadro combinado, se debe seleccionar **LISTAR** para listar los clientes registrados.
- Al final se debe mostrar el total de clientes asociados a un determinado distrito.

El objetivo de cada archivo del proyecto es:

- **conecta.php:** Contiene las sentencias que permite conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **listadoClientesxDistrito.php:** Contiene el proyecto web que interactúa con el usuario.
- **captura.php:** Contiene la función **getDistrito** que permite capturar el código del distrito seleccionado por el usuario.
- **miEstilo.css:** Contiene los estilos que se aplicarán en el archivo **listadoClientesxDistrito**.

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **listadoClientesxDistrito.php**

```
<!DOCTYPE html>
<html>
<head>
   <meta http-equiv="Content-Type"
         content="text/html; charset=ISO-8859-1" />
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">
</head>
<body>
  <header>
    <h3>LISTADO DE CLIENTES POR DISTRITO</h3>
    <img src="banner_clientes.png" width="700" height="210"/>
  </header>
  <section>
    <?php
       error_reporting(0);
       include('conecta.php');
       include('captura.php');
    ?>
    <form action="listadoClientesxDistrito.php" method="POST">
     <table border="0" width="700" cellspacing="1" cellpadding="1">
       <tr>
         <td>Seleccione Distrito</td>
         <td><select name="selDistrito">
                <?php
                   $rs=mysql_query('SELECT DISTINCT C.ID_DISTRITO,
                                           D.DESCRIPCION
                                           FROM CLIENTE C
                                           INNER JOIN DISTRITO D
                                    ON C.ID_DISTRITO = D.ID_DISTRITO',$cn);
                   $n=mysql_num_rows($rs);
                   for($i=0;$i<$n;$i++){
                     $codigo = mysql_result($rs, $i,0);
                     $distrito = mysql_result($rs, $i,1);
                     echo "<option value='".$codigo."' >"
                                           .$distrito."</option>";
                   }
                 ?>
             </select>
         </td>
         <td><input type="submit" name="btnListar" value="LISTAR" /></td>
       </tr>
     </table>
     <?php
        $codigo=getDistrito();
        $consulta=mysql_query("select descripcion
                               from distrito
                               where id_distrito='$codigo'");
        $distrito=mysql_result($consulta, 0,0);
```

```
        //Implementando la consulta
        $sql="SELECT C.ID_CLIENTE,
                     CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO),
                     C.FONO,D.DESCRIPCION
                     FROM CLIENTE C
                     INNER JOIN DISTRITO D
                     ON C.ID_DISTRITO=D.ID_DISTRITO
                     WHERE C.ID_DISTRITO='$codigo'";

            //Realizando la consulta en base al distrito seleccionado
            $rsC=mysql_query($sql,$cn);

            //Determinar el total de clientes
            $total=mysql_num_rows($rsC);
        ?>
        <p id="centrado">CLIENTES DEL DISTRITO: <?php echo $distrito; ?></p>
        <table border="0" width="700" cellspacing="1" cellpadding="1">
        <tr>
            <th>CODIGO</th>
            <th>CLIENTE</th>
            <th>TELEFONO</th>
            <th>DISTRITO</th>
        </tr>
        <?php
            for($i=0;$i<$total;$i++){
        ?>
        <tr>
            <td><?php echo mysql_result($rsC, $i, 0) ?></td>
            <td><?php echo mysql_result($rsC, $i, 1) ?></td>
            <td><?php echo mysql_result($rsC, $i, 2) ?></td>
            <td><?php echo mysql_result($rsC, $i, 3) ?></td>
        </tr>
            <?php } ?>
        <tr>
            <td></td>
            <td></td>
            <th>TOTAL</th>
            <th><?php echo $total; ?></th>
        </tr>
        </table>
    </section>
    <footer>
        <h5>Todos los derechos reservados @2015</h5>
    </footer>
</body>
</html>
```

Comentarios

```
<?php
   error_reporting(0);
   include('conecta.php');
   include('captura.php');
?>
```

Se incluye en el archivo **ListadoClientesxDistrito.php** los archivos **conecta.php** y **captura.php**. El primero realiza la conexión a la base de datos Ventas2015 y el segundo captura los valores ingresados o seleccionados por el usuario. Inicialmente bloquee los mensajes de errores de cuidado con la función **error_reporting**.

```
<?php
   $rs=mysql_query('SELECT DISTINCT C.ID_DISTRITO,
                           D.DESCRIPCION
                           FROM CLIENTE C
                           INNER JOIN DISTRITO D
                           ON C.ID_DISTRITO = D.ID_DISTRITO',$cn);
   $n=mysql_num_rows($rs);
   for($i=0;$i<$n;$i++){
     $codigo = mysql_result($rs, $i,0);
     $distrito = mysql_result($rs, $i,1);
     echo "<option value='".$codigo."' >"
                 .$distrito."</option>";
   }
?>
```

Objetivo: Llenar el cuadro combinado con el nombre del distrito y que al seleccionar un distrito este envíe su código para la búsqueda respectiva.

La variable **$rs** es la encargada de recepcionar el resultado de la consulta, para lo cual debe usarse la función **mysql_query** por tratarse de una consulta; dicha consulta usa la cláusula DISTINCT para no repetir los códigos de los distritos y el nombre del distrito.

Además, se declara la variable **n** que permite almacenar el total de registros capturados en la consulta anterior, esta será referida por medio de la variable **$rs**.

Luego se recorren los registros usando la estructura repetitiva **for** desde el registro de índice 0 hasta el último registro; cuyo valor es obtenido desde la variable **n**. Una vez dentro del ciclo de repeticiones se capturan los códigos y el nombre de los distritos por medio de las variables código y distrito, respectivamente; se usa la función **mysql_result** para la captura, dicha función usa como parámetro la variable **$rs** que indica de dónde obtendrá los datos, la variable **$i** que representa la fila de valores y 0 que representa el número de columna a obtener; este último podría ser modificado por el nombre de la columna; por ejemplo, **id_distrito**.

Finalmente, se realiza la impresión de los valores capturados usando las variables registro y distrito dentro de la etiqueta <option> para llenar el cuadro combinado con los nombres de los distritos, cuyo valor sería el código del mismo distrito.

```
<?php
   $codigo=getDistrito();
   $consulta=mysql_query("select descripcion
                                from distrito
                                where id_distrito='$codigo'");
   $distrito=mysql_result($consulta, 0,0);

   //Implementando la consulta
   $sql="SELECT C.ID_CLIENTE,
                    CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO),
                    C.FONO,D.DESCRIPCION
                    FROM CLIENTE C
                    INNER JOIN DISTRITO D
```

```
                    ON C.ID_DISTRITO=D.ID_DISTRITO
                    WHERE C.ID_DISTRITO='$codigo'";

    //Realizando la consulta en base al distrito seleccionado
    $rsC=mysql_query($sql,$cn);

    //Determinar el total de clientes
    $total=mysql_num_rows($rsC);
?>
```

Se inicia el proceso capturando el código del distrito obtenido desde la función **getDistrito()**, el cual se obtiene a su vez del cuadro combinado llenado en el paso anterior. Luego se declara la variable **$consulta** que tiene por misión obtener el nombre del distrito seleccionado a partir de la consulta realizada a la tabla Distrito, este resultado es enviado a la variable **$distrito**; posteriormente, esta variable mostrará qué distrito seleccionó el usuario de la siguiente manera:

CLIENTES DEL DISTRITO: PACHACAMAC

Seguidamente, se declara la variable **$sql** que tiene la misión de obtener los registros de los clientes a partir de una consulta a la tabla Cliente y condicionando que sean de un determinado distrito; esto se evalúa mediante el valor obtenido en la variable **$codigo**, el cual almacena el código del distrito seleccionado por el usuario.

Se ejecuta la consulta y envía el resultado a la variable **$rsC**; se le llama así porque ya existía una variable **$rs**. Finalmente, se determina el total de clientes de un determinado distrito usando la función **mysql_num_rows** y evaluando la variable **$rsC**.

```
<p id="centrado">CLIENTES DEL DISTRITO: <?php echo $distrito; ?></p>
```

La variable **$distrito** contiene el nombre del distrito seleccionado desde el cuadro combinado; este se imprimirá como cabecera en el listado de clientes por distrito.

```
<?php
    for($i=0;$i<$total;$i++){
?>
    <tr>
        <td><?php echo mysql_result($rsC, $i, 0) ?></td>
        <td><?php echo mysql_result($rsC, $i, 1) ?></td>
        <td><?php echo mysql_result($rsC, $i, 2) ?></td>
        <td><?php echo mysql_result($rsC, $i, 3) ?></td>
    </tr>
<?php } ?>
```

Para imprimir los datos de los clientes, se debe tener en cuenta que la variable **$rsC** contiene los datos de los clientes según el distrito seleccionado por el usuario, ahora se imprimirán dentro de las celdas correspondientes; para obtener dichos registros se hará un recorrido por los registros resultantes usando la estructura repetitiva **for**. Finalmente, tenga en cuenta que la variable **$rsC** contiene las columnas 0 para el código, 1 para el nombre completo, 2 para el teléfono y 3 para el nombre del distrito de donde proviene el cliente.

```
<tr>
   <td></td>
   <td></td>
   <th>TOTAL</th>
   <th><?php echo $total; ?></th>
</tr>
```

La variable **$total** permite imprimir el total de clientes encontrados por distrito. Debe tener en cuenta que en las líneas anteriores se capturó el total de clientes en la variable **$total** usando la función **mysql_num_rows**.

Archivo: **captura.php**

```
<?php
    function getDistrito(){
        return $_POST['selDistrito'];
    }
?>
```

Comentarios

```
function getDistrito(){
        return $_POST['selDistrito'];
}
```

La función **getDistrito()** tiene la misión de capturar el código del distrito seleccionado por el usuario; debe tener en cuenta que el usuario visualiza el nombre del distrito, pero al seleccionar uno de ellos es el código el que debemos capturar, ya que este permitirá buscar los datos de los clientes según el código del distrito, esto se ejecuta gracias a la especificación realizada al momento de llenar el cuadro combinado y, especialmente, al hacer referencia a la propiedad **value='".$codigo."'** del control <SELECT>.

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
#centrado{
    text-align: center;
}
table {
    margin: auto;
    background-color: antiquewhite;
}
img{
    margin: auto;
    display: block;
}
table th {
background-color: #b9c9fe;
}
```

Caso desarrollado 2 **Listado de boletas por cliente**

Implementar una aplicación web con PHP y MySQL que permita listar las boletas de un determinado cliente, tal como se muestra en la siguiente imagen:

Imagen:boletasxCliente.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **BoletasxCliente**.
- Se debe ingresar el código del cliente mediante un control de cuadro de texto, al seleccionar **CONSULTAR** se mostrarán las boletas asociadas a dicho cliente.
- La lista debe mostrar el número de la boleta, fecha de registro, código del cliente, nombre completo del cliente y el subtotal acumulado por boleta.
- Al final se debe mostrar el monto acumulado de todas las boletas del cliente.

El objetivo de cada archivo del proyecto es:

- **conecta.php:** Contiene las sentencias que permite conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **boletasxCliente.php:** Contiene el proyecto web que interactúa con el usuario.
- **captura.php:** Contiene la función **getCodigo** que permite capturar el código del cliente a buscar.
- **miEstilo.css:** Contiene los estilos que se aplicarán en el archivo **boletasxCliente**.

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **boletasxCliente.php**

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">
</head>
<body>
    <?php
        error_reporting(0);
        include('conecta.php');
        include('captura.php');
    ?>
    <header>
        <h3>LISTADO DE BOLETAS POR CLIENTE</h3>
        <img src="banner_clientes.png" width="700" height="210"/>
    </header>
    <section>
    <form method="POST" action="boletasxCliente.php">
        <table border="1" width="700" cellspacing="1" cellpadding="1">
        <tr>
            <td>INGRESE CODIGO DE CLIENTE</td>
            <td><input type="text" name="txtCodigo" maxlength="5"
                       value="<?php echo getCodigo(); ?>" />
            </td>
            <td><input type="submit" value="CONSULTAR" /></td>
        </tr>
        </table>
    </form>
    <table border="0" width="700" cellspacing="1" cellpadding="1">
    <tr>
        <th>Nº BOLETA</th>
        <th>FECHA</th>
        <th>CODIGO DE CLIENTE</th>
        <th>NOMBRE DEL CLIENTE</th>
        <th>SUBTOTAL</th>
    </tr>
    <?php
        $codigo = getCodigo();
        $sql="SELECT B.NUM_BOLETA,B.FECHA,C.ID_CLIENTE,
                CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO),
                SUM(P.PRECIO*D.CANTIDAD)
              FROM CLIENTE C
              INNER JOIN BOLETA B ON C.ID_CLIENTE = B.ID_CLIENTE
              INNER JOIN DETALLEBOLETA D ON D.NUM_BOLETA = B.NUM_BOLETA
              INNER JOIN PRODUCTO P ON P.ID_PRODUCTO = D.ID_PRODUCTO
              GROUP BY B.NUM_BOLETA,B.FECHA,C.ID_CLIENTE,
                       CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO)
              HAVING C.ID_CLIENTE='$codigo'";

        $rs=mysql_query($sql,$cn);
        $n=  mysql_num_rows($rs);

        for($i=0;$i<$n;$i++){
            //Acumulando los subtotales por cada boleta
            $acumulado+=mysql_result($rs, $i,4);
    ?>
    <tr>
      <td><?php echo mysql_result($rs, $i,0) ?></td>
      <td><?php echo mysql_result($rs, $i,1) ?></td>
```

```
        <td><?php echo mysql_result($rs, $i,2) ?></td>
        <td><?php echo mysql_result($rs, $i,3) ?></td>
        <td><?php echo '$ '.number_format(mysql_result($rs, $i,4),2) ?></td>
      </tr>
        <?php } ?>
      <tr>
          <td></td>
          <td></td>
          <td></td>
          <th>COSTO ACUMULADO</th>
          <th><?php echo "$ ".number_format($acumulado,2) ?></th>
      </tr>
      </table>
      </section>
      <footer>
          <h5>Todos los derechos reservados @2015</h5>
      </footer>
  </body>
  </html>
```

Comentarios

```
<?php
    $codigo = getCodigo();
    $sql="SELECT B.NUM_BOLETA,B.FECHA,C.ID_CLIENTE,
                CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO),
                SUM(P.PRECIO*D.CANTIDAD)
              FROM CLIENTE C
              INNER JOIN BOLETA B ON C.ID_CLIENTE = B.ID_CLIENTE
              INNER JOIN DETALLEBOLETA D ON D.NUM_BOLETA = B.NUM_BOLETA
              INNER JOIN PRODUCTO P ON P.ID_PRODUCTO = D.ID_PRODUCTO
              GROUP BY B.NUM_BOLETA,B.FECHA,C.ID_CLIENTE,
                        CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO)
              HAVING C.ID_CLIENTE='$codigo'";

    $rs=mysql_query($sql,$cn);
    $n=  mysql_num_rows($rs);

    for($i=0;$i<$n;$i++){
        //Acumulando los subtotales por cada boleta
        $acumulado+=mysql_result($rs, $i,4);
?>
```

Se captura el código obtenido a partir de la función **getCodigo()** el cual obtiene la información a partir del ingreso de código del cliente. Luego, se implementa la consulta en la variable **$sql** que permite consultar a la tabla Cliente, unida a la Boleta, detalle y producto. Ahora se estará preguntando por qué tantas uniones, la justificación de muchas uniones se debe a las columnas que se desean mostrar en el resultado. Se empezará por el número de boleta y la fecha que provienen de la tabla Boleta, para calcular el subtotal se necesita el precio del producto obtenido desde la tabla Producto y la cantidad adquirida por el cliente obtenida desde el detalle de la boleta; además, se condiciona que los datos a mostrar sean exclusivamente de un determinado cliente, esto se implementa mediante la cláusula **having c.id_cliente='$codigo'**.

El resultado de la consulta es almacenada en la variable **$rs**, desde aquí se puede desprender el total de boletas por cliente para poder imprimirlos en la tabla HTML. En el caso de la acumulación, se declaró una variable **$acumulado** que permite sumar todos los valores de la columna 4, es decir, nos referimos a los subtotales.

```
<tr>
      <td><?php echo mysql_result($rs, $i,0) ?></td>
      <td><?php echo mysql_result($rs, $i,1) ?></td>
      <td><?php echo mysql_result($rs, $i,2) ?></td>
      <td><?php echo mysql_result($rs, $i,3) ?></td>
      <td><?php echo '$ '.number_format(mysql_result($rs, $i,4),2) ?></td>
</tr>
```

Los resultados son enviados a la tabla mediante la función **mysql_result**. Esta función especifica los registros obtenidos en la variable **$rs**, los registros a mostrar con la variable **$i** y las columnas a mostrar, donde 0 representa al código de la boleta, 1 a la fecha de la boleta, 2 al código del cliente, 3 al nombre completo del cliente y 4 al subtotal obtenido por boleta. Finalmente, se usa la función **.number_format** para redondear el resultado de los subtotales a dos decimales.

```
<th><?php echo "$ ".number_format($acumulado,2) ?></th>
```

Se imprime la variable **$acumulado** a dos decimales usando la función **number_format**, considere que dicha variable posee el acumulado de los subtotales.

Archivo: **captura.php**

```
<?php
    function getCodigo(){
        return $_POST['txtCodigo'];
    }
?>
```

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
#centrado{
    text-align: center;
}
table {
    margin: auto;
    background-color: antiquewhite;
}
img{
    margin: auto;
    display: block;
}
table th {
background-color: #b9c9fe;
}
```

Caso desarrollado 3 **Listado de clientes con imágenes**

Implementar una aplicación web con PHP y MySQL que permita listar los clientes con sus respectivas imágenes, tal como se muestra en la siguiente imagen:

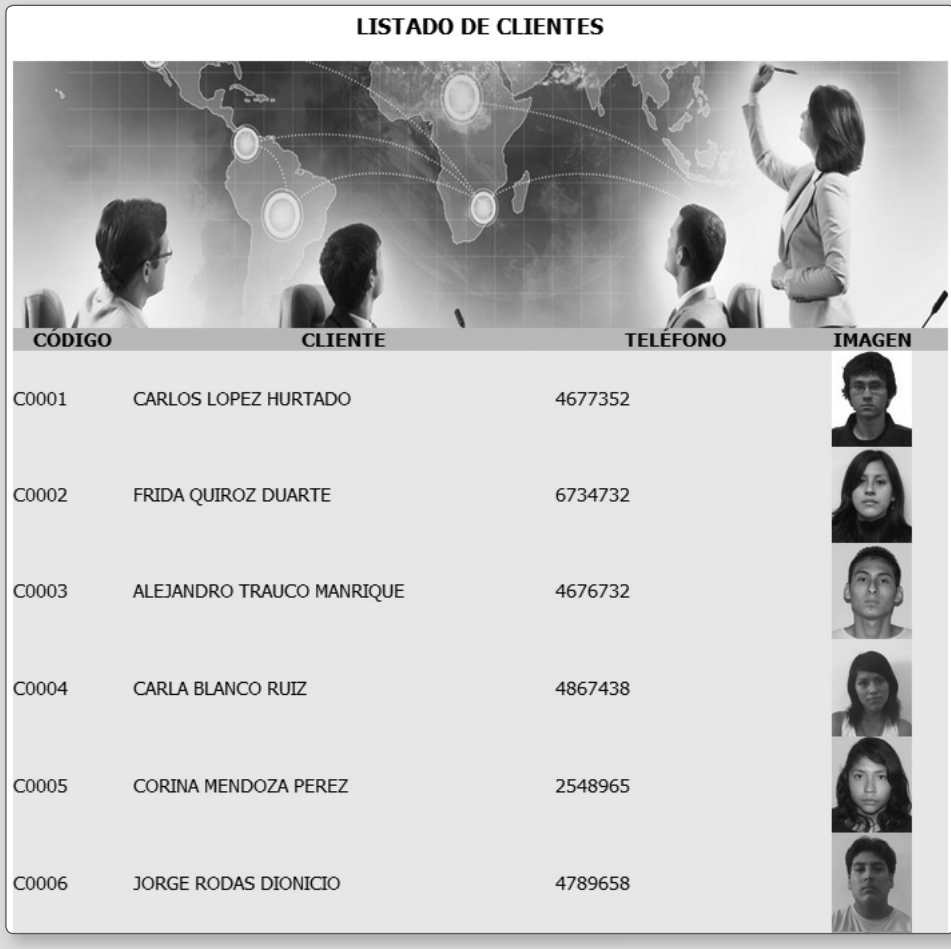


Imagen: listado.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **ImagenesClientes**.
- La lista debe mostrar el código, nombre completo del cliente, teléfono y la foto del cliente.
- Debe contar con una carpeta llamada **FOTOS** que contenga las imágenes de los clientes, tal como se muestra a continuación:

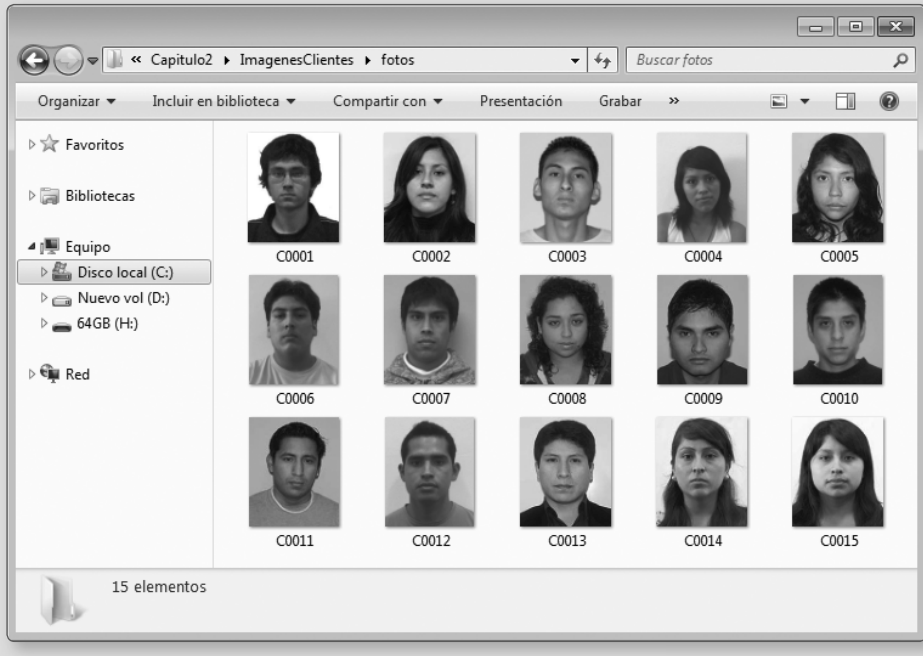


Imagen: carpeta fotos el cual contiene todas las imágenes de los clientes en formato JPG.

El objetivo de cada archivo del proyecto es:

- **conecta.php:** Contiene las sentencias que permite conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **listado.php:** Contiene el proyecto web que interactúa con el usuario.
- **miEstilo.css:** Contiene los estilos que se aplicarán en el archivo **listado**.

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **listado.php**

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">
</head>
<body>
    <?php
        include('conecta.php');
    ?>
  <header>
    <h3>LISTADO DE CLIENTES</h3>
    <img src="banner_clientes.png" width="700" height="200">
```

```
    </header>
    <section>
      <table width="700" border="0"
             cellpadding="0" cellspacing="0">
       <tr>
              <th width="80">CODIGO</th>
              <th width="280">CLIENTE</th>
              <th width="160">TELEFONO</th>
              <th width="100">IMAGEN</th>
       </tr>
       <?php
              $rs=mysql_query("SELECT ID_CLIENTE,
                                  CONCAT(NOMBRES,' ',PATERNO,' ',MATERNO),
                                  FONO FROM CLIENTE",$cn);
              $n=mysql_num_rows($rs);

              for($i=0;$i<$n;$i++){
       ?>
       <tr>
              <td><?php echo mysql_result($rs,$i,0);?></td>
              <td><?php echo mysql_result($rs,$i,1);?></td>
              <td><?php echo mysql_result($rs,$i,2);?></td>
              <td>
              <?php
                  echo "<img src=fotos/".mysql_result($rs,$i,0).".jpg
                             width=60>";
              ?>
              </td>
       </tr>
       <?php
              }
       ?>
       </table>
  </section>
  <footer>
      <h5>Todos los derechos reservados @2015</h5>
  </footer>
  </body>
  </html>
```

Comentarios

```
<?php
   include('conecta.php');
?>
```

Se incluye en el archivo **listado.php** las sentencias implementadas en el archivo **conecta.php**, esto permitirá conectarse a la base de datos Ventas2015, de forma que se puede consultar los registros desde las tablas.

```
<?php
  $rs=mysql_query("SELECT ID_CLIENTE,
                          CONCAT(NOMBRES,' ',PATERNO,' ',MATERNO),
                          FONO FROM CLIENTE",$cn);
  $n=mysql_num_rows($rs);

  for($i=0;$i<$n;$i++){
?>
```

La variable **$rs** permite obtener el resultado de la consulta realizada a la tabla Cliente. Debe considerar que los nombres, apellidos deben generar una sola columna; esto se realiza mediante la función **concat** de mysql. La variable **$n** obtiene el número total de clientes, esto ayudará a recorrer de manera exacta por la cantidad de registros resultantes de la consulta.

```
<tr>
  <td><?php echo mysql_result($rs,$i,0);?></td>
  <td><?php echo mysql_result($rs,$i,1);?></td>
  <td><?php echo mysql_result($rs,$i,2);?></td>
  <td>
    <?php echo "<img src=fotos/".mysql_result($rs,$i,0).".jpg width=60>";?>
  </td>
</tr>
```

La variable **$rs** contiene todos los registros resultantes que se quiere mostrar, se realizará dicha impresión usando la función **mysql_result**, considere que **$i** accede a cada fila de los registros y el número 0 pertenece al código, 1 a los nombres completos y 2 al teléfono de los clientes. Finalmente, la impresión de la imagen se realiza mediante la etiqueta **<IMG>** y especificando en la propiedad SCR el lugar y el nombre exacto de la imagen; aquí, debe recordar que el nombre de la imagen es la misma que el código del cliente, en la propiedad width se define el ancho de la imagen, el cual podría ser modificado a conveniencia propia.

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
table {
    margin: auto;
    background-color: antiquewhite;
}
img{
    margin: auto;
    display: block;
}
table th {
background-color: #b9c9fe;
}
```

Caso desarrollado 4 **Listado de clientes paginado con imágenes**

Implementar una aplicación web con PHP y MySQL que permita listar los clientes de forma paginada, tal como se muestra en la siguiente imagen:

Imagen: paginacion.php mostrando los registros de la página 1.

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **ImagenesClientes**.
- Cada página solo debe mostrar cuatro clientes y tener un enlace para dirigirse a los demás registros.
- La lista debe mostrar el código, nombre completo del cliente, teléfono y la foto del cliente.
- Debe contar con una carpeta llamada **FOTOS** que contenga las imágenes de los clientes, tal como se muestra a continuación:

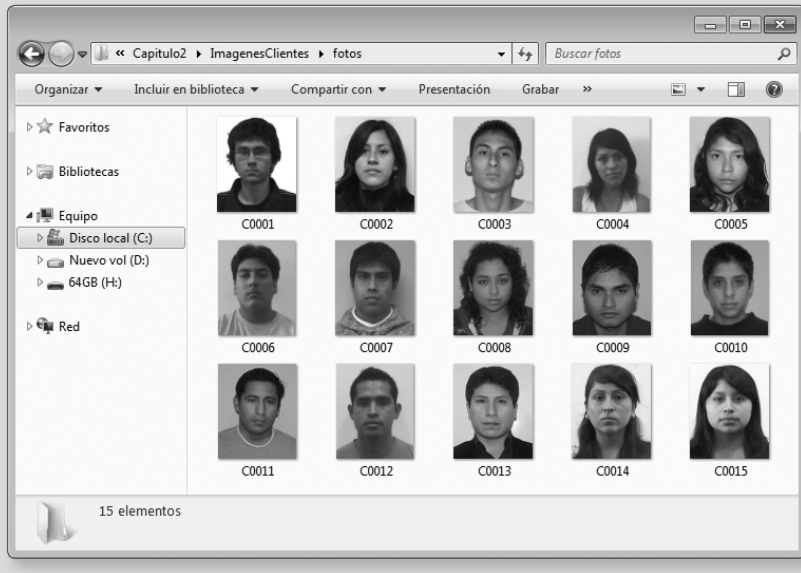


Imagen: carpeta fotos el cual contiene todas las imágenes de los clientes en formato JPG.

- Al presionar en un número de página, se deben mostrar los registros correspondientes, además de mover el link al número siguiente de página; es decir, si se encuentra visualizando la página 2, esta no tendrá enlace, tal como se muestra en las siguientes imágenes:

Imagen: paginacion.php mostrando los registros de la página 2

Imagen: paginacion.php mostrando los registros de la página 3

Imagen: paginacion.php mostrando los registros de la página 4

El objetivo de cada archivo del proyecto es:

- **conecta.php**: Contiene las sentencias que permite conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **paginación.php**: Contiene el proyecto web que interactúa con el usuario.
- **miEstilo.css**: Contiene los estilos que se aplicarán en el archivo **listadoClientesxDistrito**.

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **paginación.php**

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">
</head>
<body>
    <?php
        include('conecta.php');
    ?>
  <header>
    <h3>LISTADO DE CLIENTES</h3>
    <img src="banner_clientes.png" width="700" height="200">
  </header>
  <section>
    <table width="700" border="0"
           cellpadding="0" cellspacing="0">
    <tr>
            <th width="80">CODIGO</th>
            <th width="280">CLIENTE</th>
            <th width="160">TELEFONO</th>
            <th width="100">IMAGEN</th>
    </tr>
      <?php
        //Asignando el número de registros a mostrar
        $filas=4;

        //Evaluar el número de página(nPag)
        //Si la variable nPag esta vacía la página inicial será 0
        //caso contrario se captura la pagina seleccionada por el usuario.
        if (!isset($_GET["nPag"])) $nPag=0; else $nPag=$_GET["nPag"];

        //La variable inicio asigna el valor para cada juego de registros
        //1..4  Inicio=0
        //5..8  Inicio=4
        //9..12 Inicio=8
        $inicio=$nPag*$filas;

        //Obteniendo los registros a mostrar
        $rs=mysql_query("SELECT ID_CLIENTE,
                                CONCAT(NOMBRES,' ',PATERNO,' ',MATERNO),
                                FONO FROM CLIENTE LIMIT $inicio,$filas",$cn);
```

```
        //Determinar el total de clientes
        $n=mysql_num_rows($rs);

        //Recorrer por los registros
        for($i=0;$i<$n;$i++){
    ?>
    <tr>
        <td><?php echo mysql_result($rs,$i,0);?></td>
        <td><?php echo mysql_result($rs,$i,1);?></td>
        <td><?php echo mysql_result($rs,$i,2);?></td>
        <td>
          <?php
             //Mostrando la imagen del cliente
             echo "<img src=fotos/".mysql_result($rs,$i,0).".jpg width=60>";
          ?>
        </td>
    </tr>
        <?php
            }
      ?>
      <tr>
        <th colspan="4">
           <?php
              //Implementar la consulta de clientes
              $rsC=mysql_query("SELECT * FROM CLIENTE",$cn);

              //Determinar el total de registros
              $num=mysql_num_rows($rsC);

              //Determinar el total de paginas
              //(Total de Registros/Registros a mostrar)
              $totalPag=floor($num/$filas);

              //Si en la última hoja solo hay un registro este genera una
              //nueva pagina aumentando así el total de paginas.
              if(($num%$filas)>0) $totalPag++;

              //Imprimiendo el número actual y el total de páginas
              echo "Pagina Nº ".($nPag+1)." de ".$totalPag;
           ?>

           <br>
           <?php
              //Generando el enlace a las demás hojas de la paginación
              for($i=0;$i<$totalPag;$i++){
                    if ($i==$nPag){
                         echo "[".($i+1)."]";
                      } else {
           ?>
           <a href='<?php echo $_SERVER['PHP_SELF']."?nPag=$i"?>'>
                    <?php echo "[".($i+1)."]" ?> </a>
           <?php
                   } //Cierre de else
             } //Cierre del for
           ?>
        </th>
    </tr>
    </table>
</section>
<footer>
```

```
    <h5>Todos los derechos reservados @2015</h5>
</footer>
</body>
</html>
```

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
#centrado{
    text-align: center;
}
table {
    margin: auto;
    background-color: antiquewhite;
}
img{
    margin: auto;
    display: block;
}
table th {
background-color: #b9c9fe;
}
```

2.2.8 Manejo de registros usando arreglos de índice

A. Comentario

Para obtener los registros a partir de una consulta implementada con la función **mysql_query**, se usó la función **mysql_result**, que permitía devolver los registros a una variable que a su vez era descargada en una tabla. Ahora se verá cómo transferir dicha información a un arreglo de tipo índice usando la función **mysql_fetch_row**.

B. Sintaxis

```
$nombre_arreglo_indice = mysql_fetch_row($variable_de_consulta)
```

La sentencia debe ser parte de la estructura repetitiva **for** o **while** solo si hay varios registros que obtener desde la consulta.

- $nombre_arreglo_indice: Es la variable que contendrá todos los registros de la consulta.
- $variable_de_consulta: Es la especificación a la consulta realizada con la función **mysql_query**.

Caso 1: Implementar una aplicación en PHP que permita listar los clientes de la base de datos Ventas2015 usando arreglos por índice, tal como se muestra en la siguiente imagen:

LISTADO DE CLIENTES

USANDO ARREGLO POR ÍNDICE

CÓDIGO	CLIENTE	DISTRITO
C0001	CARLOS LOPEZ HURTADO	CERCADO
C0002	FRIDA QUIROZ DUARTE	CARABAYLLO
C0003	ALEJANDRO TRAUCO MANRIQUE	BARRANCO
C0004	CARLA BLANCO RUIZ	ATE
C0005	CORINA MENDOZA PEREZ	CHORRILLOS
C0006	JORGE RODAS DIONICIO	ATE
C0007	ORLANDO CUEVAS CABANILLAS	JESUS MARIA
C0008	ANTUANE RODRIGUEZ ALARCON	BREÑA
C0009	MANUEL SUAREZ FERNANDEZ	PACHACAMAC
C0010	JUAN GUTIERREZ DIAZ	LINCE
C0011	CARLOS COLAN BARDALES	PACHACAMAC
C0012	MARTIN CARRILLO SALAS	CHACLACAYO
C0013	JOSE LAZARTE LUJAN	MAGDALENA
C0014	VIDAL ZORRILLA RODRIGUEZ	PUCUSANA
C0015	GUILLERMO RAMOS FLORES	ATE

EL TOTAL DE CLIENTES ES: 15

Todos los derechos reservados @2015

Imagen: listado3.php

Archivo: **listado3.php**

```
<!DOCTYPE html>
<html>
 <head>
   <meta http-equiv="Content-Type" content="text/html;
         charset=ISO-8859-1" />
   <title></title>
   <link href="miEstilo.css" rel="stylesheet">
 </head>
 <body>
   <header>
     <img src="banner_clientes.png" width="940" height="210"/>
<h2>LISTADO DE CLIENTES</h2>
<h5>USANDO ARREGLO POR INDICE</h5>
   </header>
   <section>
<?php
   /* Conexión al servidor */
   $cn = mysql_connect('localhost','root','');

   /* Seleccionando la base de datos Ventas2015 */
   mysql_select_db('ventas2015',$cn);

   /* Consultando la tabla cliente */
   $rs = mysql_query("SELECT C.ID_CLIENTE,
              CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO),
              D.DESCRIPCION
              FROM CLIENTE C
              INNER JOIN DISTRITO D
              ON C.ID_DISTRITO=D.ID_DISTRITO",$cn);

   /* Determinar el total de clientes */
   $n = mysql_num_rows($rs);
?>
```

```
<table border="0" width="650" cellspacing="0" cellpadding="0">
  <tr>
    <td id="fila">CODIGO</td>
    <td id="fila">CLIENTE</td>
    <td id="fila">DISTRITO</td>
  </tr>
  <?php
     while($clientes = mysql_fetch_row($rs)){
  ?>
  <tr>
    <td><?php echo $clientes[0];?></td>
    <td><?php echo $clientes[1];?></td>
    <td><?php echo $clientes[2];?></td>
  </tr>
  <?php } ?>
  <tr>
    <td colspan="6">
         <?php echo 'EL TOTAL DE CLIENTES ES: '.$n; ?>
    </td>
  </tr>
 </table>
    </section>
    <footer>
  <h4>Todos los derechos reservados @2015</h4>
    </footer>
 </body>
</html>
```

Archivo: **miEstilo.css**

```
body {
    font-family: tahoma;
    font-size: 12px;
}
table {
   width: 100%;
   border: 1px solid #000;
}
#fila {
   background: activecaption;
   font-size: 14px;
}
```

2.2.9 Manejo de registros usando arreglo asociativo

A. Comentario

De la misma forma con que se manejaron los registros usando arreglos de índice, se pueden transferir registros a un arreglo asociativo.

B. Sintaxis

```
$nombre_arreglo_asociativo = mysql_fetch_array($variable_de_consulta)
```

La sentencia debe ser parte de la estructura repetitiva **for** o **while** solo si hay varios registros que obtener desde la consulta.

- $nombre_arreglo_asociativo: Es la variable que contendrá todos los registros de la consulta.
- $variable_de_consulta: Es la especificación a la consulta realizada con la función **mysql_query**.

Caso 1: Implementar una aplicación en PHP que permita listar los clientes de la base de datos Ventas2015 usando arreglos asociativos, tal como se muestra en la siguiente imagen:

LISTADO DE CLIENTES

USANDO ARREGLO ASOCIATIVO

CÓDIGO	CLIENTE	DISTRITO
C0001	CARLOS LOPEZ HURTADO	CERCADO
C0002	FRIDA QUIROZ DUARTE	CARABAYLLO
C0003	ALEJANDRO TRAUCO MANRIQUE	BARRANCO
C0004	CARLA BLANCO RUIZ	ATE
C0005	CORINA MENDOZA PEREZ	CHORRILLOS
C0006	JORGE RODAS DIONICIO	ATE
C0007	ORLANDO CUEVAS CABANILLAS	JESUS MARIA
C0008	ANTUANE RODRIGUEZ ALARCON	BREÑA
C0009	MANUEL SUAREZ FERNANDEZ	PACHACAMAC
C0010	JUAN GUTIERREZ DIAZ	LINCE
C0011	CARLOS COLAN BARDALES	PACHACAMAC
C0012	MARTIN CARRILLO SALAS	CHACLACAYO
C0013	JOSE LAZARTE LUJAN	MAGDALENA
C0014	VIDAL ZORRILLA RODRIGUEZ	PUCUSANA
C0015	GUILLERMO RAMOS FLORES	ATE

EL TOTAL DE CLIENTES ES: 15

Todos los derechos reservados @2015

Imagen: listado4.php

Archivo: **listado4.php**

```
<!DOCTYPE html>
<html>
 <head>
   <meta http-equiv="Content-Type" content="text/html;
         charset=ISO-8859-1" />
   <title></title>
   <link href="miEstilo.css" rel="stylesheet">
 </head>
 <body>
     <header>
         <img src="banner_clientes.png"
              width="940" height="210"/>
         <h2>LISTADO DE CLIENTES</h2>
         <h5>USANDO ARREGLO ASOCIATIVO</h5>
   </header>
   <section>
     <?php
        /* Conexión al servidor */
        $cn = mysql_connect('localhost','root','');

        /* Seleccionando la base de datos Ventas2015 */
        mysql_select_db('ventas2015',$cn);

        /* Consultando la tabla cliente */
        $rs = mysql_query("SELECT C.ID_CLIENTE,
             CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
```

```
                D.DESCRIPCION FROM CLIENTE C
                INNER JOIN DISTRITO D
                ON C.ID_DISTRITO=D.ID_DISTRITO",$cn);

            /* Determinar el total de clientes */
            $n = mysql_num_rows($rs);
        ?>
        <table border="0" width="650" cellspacing="0" cellpadding="0">
          <tr>
            <td id="fila">CODIGO</td>
            <td id="fila">CLIENTE</td>
            <td id="fila">DISTRITO</td>
          </tr>
          <?php
               while($clientes = mysql_fetch_array($rs)){
          ?>
          <tr>
            <td><?php echo $clientes["ID_CLIENTE"];?></td>
            <td><?php echo $clientes["CLIENTE"];?></td>
            <td><?php echo $clientes["DESCRIPCION"];?></td>
           </tr>
           <?php } ?>
           <tr>
               <td colspan="6">
                   <?php echo 'EL TOTAL DE CLIENTES ES: '.$n; ?>
               </td>
           </tr>
           </table>
           </section>
           <footer>
               <h4>Todos los derechos reservados @2015</h4>
           </footer>
      </body>
</html>
```

Tener en cuenta lo siguiente:

- Todas las columnas de la consulta deben tener un nombre.
- La especificación de un elemento asociado diferencia entre mayúsculas y minúsculas.

Archivo: **miEstilo.css**

```
body {
    font-family: tahoma;
    font-size: 12px;
}
table {
   width: 100%;
   border: 1px solid #000;
}
#fila {
   background: activecaption;
   font-size: 14px;
}
```

2.2.10 Manejo de registros usando arreglos asociativos y foreach

A. Comentario

El uso del arreglo asociativo conlleva a que los registros consultados se almacenen en un arreglo común, desde el cual podemos obtener la información e imprimirlas, esta vez se usará la estructura repetitiva **foreach** para la impresión de los resultados. Se debe tener en cuenta que para dicha impresión primero debe estar lleno el arreglo.

B. Sintaxis

```
while ($variable_arreglo_asociativo =  mysql_fetch_array($variable_de_consulta)){
$arreglo[]= $variable_arreglo_asociativo;
}

foreach ($arreglo as $variable_arreglo_asociativo){
          echo $variable_arreglo_asociativo[nombre_columna];
}
```

La sentencia **while** se encarga de llenar el arreglo con los registros obtenidos en la consulta ejecutada por la función **mysql_query**. Mientras que **foreach** recorre por todo el arreglo para su impresión; no teniendo relación con la consulta solo con el arreglo.

- $variable_arreglo_asociativo: Es la variable que contendrá todos los registros de la consulta.
- $variable_de_consulta: Es la especificación de la consulta realizada con la función **mysql_query.**
- $arreglo[ ]: Registra el resultado de la consulta.
- nombre_columna: Es la especificación de las columnas a mostrar en el listado.

Caso 1: Implementar una aplicación en PHP que permita listar los clientes de la base de datos Ventas2015 usando arreglo asociativo y la estructura **foreach**, tal como se muestra en la siguiente imagen:

LISTADO DE CLIENTES

USANDO ARREGLO ASOCIATIVO Y FOREACH

CÓDIGO	CLIENTE	DISTRITO
C0001	CARLOS LOPEZ HURTADO	CERCADO
C0002	FRIDA QUIROZ DUARTE	CARABAYLLO
C0003	ALEJANDRO TRAUCO MANRIQUE	BARRANCO
C0004	CARLA BLANCO RUIZ	ATE
C0005	CORINA MENDOZA PEREZ	CHORRILLOS
C0006	JORGE RODAS DIONICIO	ATE
C0007	ORLANDO CUEVAS CABANILLAS	JESUS MARIA
C0008	ANTUANE RODRIGUEZ ALARCON	BREÑA
C0009	MANUEL SUAREZ FERNANDEZ	PACHACAMAC
C0010	JUAN GUTIERREZ DIAZ	LINCE
C0011	CARLOS COLAN BARDALES	PACHACAMAC
C0012	MARTIN CARRILLO SALAS	CHACLACAYO
C0013	JOSE LAZARTE LUJAN	MAGDALENA
C0014	VIDAL ZORRILLA RODRIGUEZ	PUCUSANA
C0015	GUILLERMO RAMOS FLORES	ATE

EL TOTAL DE CLIENTES ES: 15

Todos los derechos reservados @2015

Imagen: listado5.php

Archivo: **listado5.php**

```
<!DOCTYPE html>
<html>
 <head>
   <meta http-equiv="Content-Type" content="text/html;
         charset=ISO-8859-1" />
   <title></title>
   <link href="miEstilo.css" rel="stylesheet">
 </head>
 <body>
    <header>
         <img src="banner_clientes.png" width="940" height="210"/>
   <h2>LISTADO DE CLIENTES</h2>
   <h5>USANDO ARREGLO ASOCIATIVO Y FOREACH</h5>
    </header>
    <section>
<?php
  /* Conexión al servidor */
  $cn = mysql_connect('localhost','root','');

  /* Seleccionando la base de datos Ventas2015 */
  mysql_select_db('ventas2015',$cn);

  /* Consultando la tabla cliente */
  $rs = mysql_query("SELECT C.ID_CLIENTE,
           CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO),
           D.DESCRIPCION
           FROM CLIENTE C
           INNER JOIN DISTRITO D
           ON C.ID_DISTRITO=D.ID_DISTRITO",$cn);

  /* Determinar el total de clientes */
  $n = mysql_num_rows($rs);
?>
<table border="0" width="650" cellspacing="0" cellpadding="0">
 <tr>
   <td id="fila">CODIGO</td>
   <td id="fila">CLIENTE</td>
   <td id="fila">DISTRITO</td>
 </tr>
 <?php
    while ($cliente=  mysql_fetch_array($rs)){
         $misClientes[]= $cliente;
    }

    foreach($misClientes as $cliente){
 ?>
 <tr>
   <td><?php echo $cliente[0];?></td>
   <td><?php echo $cliente[1];?></td>
   <td><?php echo $cliente[2];?></td>
 </tr>
 <?php
    }
 ?>
<tr>
   <td colspan="6">
         <?php echo 'EL TOTAL DE CLIENTES ES: '.$n; ?>
   </td>
</tr>
     </table>
```

```
    </section>
    <footer>
  <h4>Todos los derechos reservados @2015</h4>
    </footer>
</body>
</html>
```

Archivo: **miEstilo.css**

```
body {
    font-family: tahoma;
    font-size: 12px;
}
table {
   width: 100%;
   border: 1px solid #000;
}
#fila {
   background: activecaption;
   font-size: 14px;
}
```

2.2.11 Manejo de registros usando procedimientos almacenados

A. Comentario

Conforme se avanza en la integración de PHP y MySQL, se van implementando consultas muy extensas que llevan a cometer ciertos errores de precisión como una coma, una comilla simple o doble; para evitarlos se usará de aquí en adelante los procedimientos almacenados de MySQL.

B. Sintaxis

Primero: Crear el procedimiento almacenado en MySQL.

```
DROP PROCEDURE IF EXISTS nombre_procedimiento;
CREATE PROCEDURE nombre_procedimiento(parámetros)
   Sentencia MySQL;
```

Se debe tener en cuenta que el punto y coma determina el fin de la implementación del procedimiento almacenado.

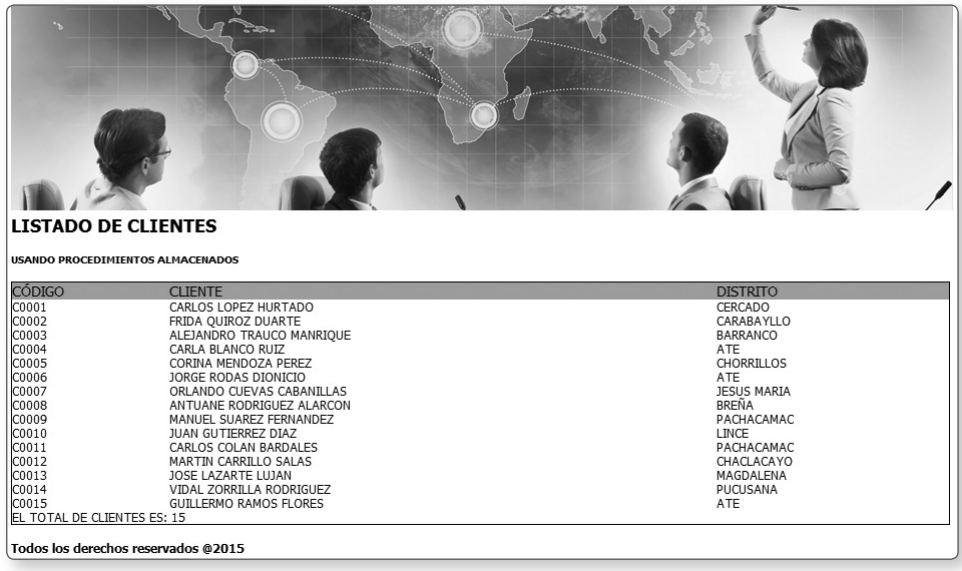
Segundo: Llamar a un procedimiento almacenado desde MySQL.

```
CALL nombre_procedimiento(parámetros);
```

Tercero: Integrar un procedimiento almacenado en una consulta desde PHP.

```
$variable_consulta=mysql_query('nombre_procedimiento',$variable_de_conexion);
```

Caso 1: Implementar una aplicación en PHP que permita listar los clientes de la base de datos Venta2015 usando procedimiento almacenado, tal como se muestra en la siguiente imagen:



LISTADO DE CLIENTES

USANDO PROCEDIMIENTOS ALMACENADOS

CÓDIGO	CLIENTE	DISTRITO
C0001	CARLOS LOPEZ HURTADO	CERCADO
C0002	FRIDA QUIROZ DUARTE	CARABAYLLO
C0003	ALEJANDRO TRAUCO MANRIQUE	BARRANCO
C0004	CARLA BLANCO RUIZ	ATE
C0005	CORINA MENDOZA PEREZ	CHORRILLOS
C0006	JORGE RODAS DIONICIO	ATE
C0007	ORLANDO CUEVAS CABANILLAS	JESUS MARIA
C0008	ANTUANE RODRIGUEZ ALARCON	BREÑA
C0009	MANUEL SUAREZ FERNANDEZ	PACHACAMAC
C0010	JUAN GUTIERREZ DIAZ	LINCE
C0011	CARLOS COLAN BARDALES	PACHACAMAC
C0012	MARTIN CARRILLO SALAS	CHACLACAYO
C0013	JOSE LAZARTE LUJAN	MAGDALENA
C0014	VIDAL ZORRILLA RODRIGUEZ	PUCUSANA
C0015	GUILLERMO RAMOS FLORES	ATE

EL TOTAL DE CLIENTES ES: 15

Todos los derechos reservados @2015

Imagen: listado6.php

En MySQL implemente el siguiente procedimiento almacenado:

```
DROP PROCEDURE IF EXISTS sp_listadoclientes;
CREATE PROCEDURE sp_listadoclientes()
    SELECT C.ID_CLIENTE,
           CONCAT(C.NOMBRES,'   ',C.PATERNO,'   ',C.MATERNO)   AS
CLIENTE,
           D.DESCRIPCION
           FROM CLIENTE C
           INNER JOIN DISTRITO D
           ON C.ID_DISTRITO=D.ID_DISTRITO;
```

Archivo: **listado6.php**

```
<!DOCTYPE html>
<html>
 <head>
   <meta http-equiv="Content-Type" content="text/html;
         charset=ISO-8859-1" />
   <title></title>
   <link href="miEstilo.css" rel="stylesheet">
 </head>
 <body>
    <header>
<img src="banner_clientes.png" width="940" height="210"/>
<h2>LISTADO DE CLIENTES</h2>
         <h5>USANDO PROCEDIMIENTOS ALMACENADOS</h5>
    </header>
    <section>
<?php
  /* Conexión al servidor */
  $cn = mysql_connect('localhost','root','');
```

```
    /* Seleccionando la base de datos Ventas2015 */
    mysql_select_db('ventas2015',$cn);

    /* Consultando la tabla cliente */
    $rs = mysql_query("CALL sp_listadoclientes",$cn);

    /* Determinar el total de clientes */
    $n = mysql_num_rows($rs);
  ?>
  <table border="0" width="650" cellspacing="0" cellpadding="0">
    <tr>
      <td id="fila">CODIGO</td>
      <td id="fila">CLIENTE</td>
      <td id="fila">DISTRITO</td>
    </tr>
    <?php
        while($clientes = mysql_fetch_array($rs)){
    ?>
   <tr>
     <td><?php echo $clientes[0];?></td>
     <td><?php echo $clientes[1];?></td>
     <td><?php echo $clientes[2];?></td>
   </tr>
   <?php } ?>
   <tr>
     <td colspan="6">
          <?php echo 'EL TOTAL DE CLIENTES ES: '.$n; ?>
     </td>
   </tr>
  </table>
      </section>
      <footer>
  <h4>Todos los derechos reservados @2015</h4>
      </footer>
  </body>
  </html>
```

Archivo: **miEstilo.css**

```
body {
    font-family: tahoma;
    font-size: 12px;
}
table {
   width: 100%;
   border: 1px solid #000;
}
#fila {
   background: activecaption;
   font-size: 14px;
}
```

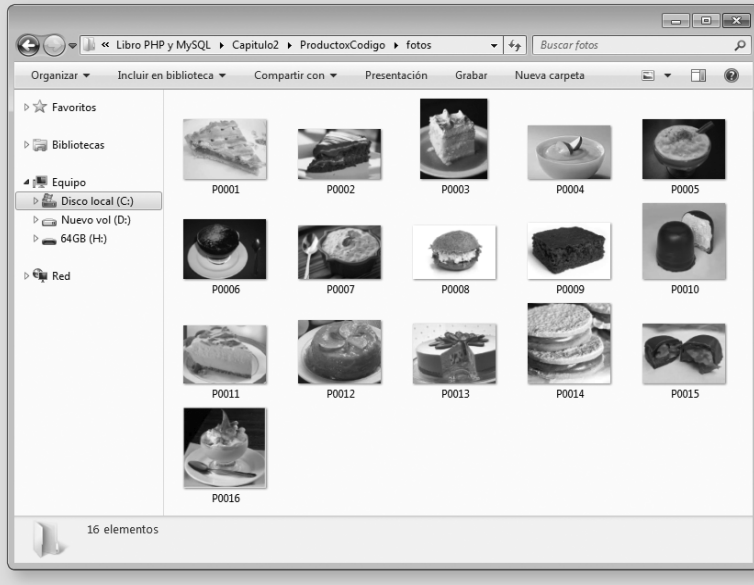
Caso desarrollado 5 **Consulta de productos por código**

Implementar una aplicación web con PHP y MySQL que permita listar los datos de un determinado producto según el código del mismo, tal como se muestra en la siguiente imagen:

Imagen: filtro.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **ProductoxCodigo**.
- El código del producto a buscar debe ingresarse en una caja de texto.
- La búsqueda debe mostrar la descripción, precio, stock, fecha de vencimiento y la imagen del producto.
- Debe contar con una carpeta llamada **FOTOS** que contenga las imágenes de los productos, tal como se muestra a continuación:



El objetivo de cada archivo del proyecto es:

- **conecta.php**: Contiene las sentencias que permiten conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **captura.php**: Contiene el método **getCodigo** que permite capturar el código del producto a buscar.
- **filtro.php**: Contiene el proyecto web que interactúa con el usuario.
- **miEstilo.css**: Contiene los estilos que se aplicarán en el archivo **filtro**.

Procedimiento almacenado

```
DROP PROCEDURE IF EXISTS sp_productoxcodigo;
CREATE PROCEDURE sp_productoxcodigo(cod char(5))
     SELECT *
              FROM PRODUCTO
              WHERE ID_PRODUCTO=cod;
```

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **captura.php**

```
<?php
    //Método que obtiene el valor ingresado en el código de producto
    function getCodigo(){
        return $_POST['txtCodigo'];
    }
?>
```

Archivo: **filtro.php**

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>CONSULTA DE PRODUCTOS</title>
        <link href="miEstilo.css" rel="stylesheet">
    </head>
    <body>
        <header>
            <h3>CONSULTA DE PRODUCTOS</h3>
            <img src="banner.jpg" width="600" height="150"/>
        </header>
        <section>
            <?php
                error_reporting(0);
                include('conexion.php');
                include('captura.php');
            ?>
            <form method="POST">
```

```
            <table border="1" width="600"
                   cellspacing="5" cellpadding="0">
                <tr>
                    <td>INGRESE CODIGO DE PRODUCTO</td>
                    <td><input type="text" name="txtCodigo"
                               value="<?php echo getCodigo(); ?>"
                               maxlength="5"/>
                    </td>
                    <td><input type="submit" value="BUSCAR" /></td>
                </tr>
            </table>
            </form>

            <?php
                $codigo=  getCodigo();
                $rs=mysql_query("CALL sp_productoxcodigo('$codigo')",$cn);
                $producto = mysql_fetch_array($rs);
            ?>
            <br>
            <table border="0" width="600" cellspacing="0" cellpadding="5">
                <tr>
                    <th>PRODUCTO</th>
                    <td width="200"><?php echo $producto[1]; ?></td>
                    <td rowspan="4">
                        <img src="fotos/<?php echo $producto[0]; ?>.jpg"
                             width="150" height="130"/>
                    </td>
                </tr>
                <tr>
                    <th>PRECIO</th>
                    <td><?php echo $producto[2]; ?></td>
                </tr>
                <tr>
                    <th>STOCK</th>
                    <td><?php echo $producto[3]; ?></td>
                </tr>
                <tr>
                    <th>FECHA DE VENCIMIENTO</th>
                    <td><?php echo $producto[4]; ?></td>
                </tr>
            </table>
        </section>
        <footer>
            <h5>Todos los derechos reservados @2015</h5>
        </footer>
    </body>
</html>
```

Comentarios

```
<?php
 error_reporting(0);
 include('conexion.php');
 include('captura.php');
?>
```

Se ocultan los errores de cuidado con la función **error_reporting**, a la vez se incluye el archivo **conexión.php** para conectarse con la base de datos y el archivo **captura.php** que permite incluir la función **getCodigo** que captura el código de producto ingresado por el usuario.

```
<input type="text" name="txtCodigo"
       value="<?php echo getCodigo(); ?>"
       maxlength="5"/>
```

Se imprime el código ingresado por el usuario en el mismo control, esto permite mantener la información registrada después del **submit** realizado por el botón **Buscar**, de otra manera al intentar buscar un determinado código, este se borrará y solo mostrará los datos restantes.

```
<?php
 $codigo=  getCodigo();
 $rs=mysql_query("CALL sp_productoxcodigo('$codigo')",$cn);
 $producto = mysql_fetch_array($rs);
?>
```

Se declara la variable **$codigo** que almacena el código obtenido desde la función **getCodigo**. Para ejecutar el procedimiento almacenado **sp_productoxcodigo** se usa la función **mysql_query** y se almacenan los registros en la variable **$rs**. Hay que tener en cuenta que el procedimiento almacenado necesita un parámetro para la búsqueda, el cual es enviado por la variable código; considere que debe colocarse entre comillas simples, ya que el código del producto es de tipo **char** en la base de datos. Finalmente, se descargan los registros en la variable de tipo arreglo **$producto** mediante la función **mysql_fetch_array**.

```
<th>PRODUCTO</th>
   <td width="200"><?php echo $producto[1]; ?>
</td>
```

En la celda correspondiente se imprime el nombre del producto mediante el arreglo **$producto**, el índice 1 pertenece a la descripción del producto.

```
<td rowspan="4">
   <img src="fotos/<?php echo $producto[0]; ?>.jpg"
        width="150" height="130"/>
</td>
```

Para imprimir la imagen del producto, se debe considerar que el nombre de la imagen tiene asignado el código del producto. Tenga en cuenta que las imágenes de los productos se encuentran en la carpeta **Fotos** y que el índice 0 desde el arreglo equivale al código del producto.

Mantenimiento de registros

3.1 Inserción de registros

A. Comentario

La inserción de nuevos registros en PHP permite registrar valores ingresados por el usuario mediante formularios HTML, estos deberán ser capturados y enviados a la tabla de una conexión activa.

B. Sintaxis

Sintaxis para la implementación del procedimiento almacenado que registra valores en una tabla de MySQL:

```
CREATE PROCEDURE sp_nombre_procedimiento(parametros)
  INSERT INTO tabla VALUES (valores);
```

Sintaxis para la llamada del procedimiento almacenado en PHP:

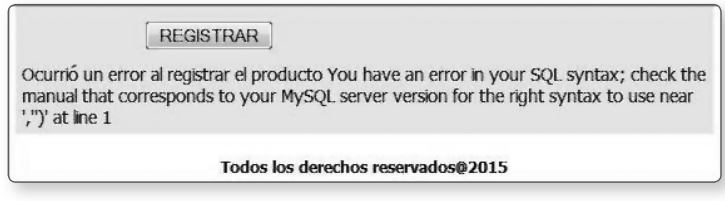
```
$variable = mysql_query('CALL sp_nombre_procedimiento(parametros)',$cn);
```

Sintaxis para la implementación de la sentencia de inserción sin procedimiento almacenado:

```
$variable = mysql_query('INSERT INTO tabla VALUES (valores) ',$cn);
```

Caso 1: Implementar una aplicación en PHP que permita registrar los datos de los productos de forma básica usando la sentencia INSERT INTO directamente en la aplicación.

En caso de que ocurra un error en el registro, se debe mostrar un mensaje al final de la siguiente forma:



En caso contrario, si el registro de producto es correcto, debe mostrar el siguiente mensaje:

Archivo: **registro1.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">
  </head>
  <body>
    <header>
      <h3>REGISTRO DE PRODUCTOS</h3>
    </header>
      <section>
        <form method="POST" action="registro.php">
          <table border="0" width="550"
                 cellspacing="0" cellpadding="5">
           <tr>
             <td>CODIGO</td>
             <td><input type="text" name="txtCodigo"
                        value="" /></td>
           </tr>
           <tr>
             <td>DESCRIPCION</td>
             <td><input type="text" name="txtDescripcion" size="70"
                        value="" /></td>
           </tr>
           <tr>
             <td>PRECIO</td>
             <td><input type="text" name="txtPrecio"
                        value="" /></td>
           </tr>
           <tr>
             <td>STOCK</td>
             <td><input type="text" name="txtStock"
                        value="" /></td>
           </tr>
           <tr>
             <td>FECHA DE VENC.</td>
             <td><input type="text" name="txtFecha"
                        value="" /></td>
           </tr>
           <tr>
             <td></td>
             <td><input type="submit" name="btnRegistrar"
                        value="REGISTRAR" /></td>
           </tr>
           <tr>
             <td colspan="2">
             <?php
              if (isset($_POST['btnRegistrar'])){
                //Capturando datos
                $codigo = $_POST['txtCodigo'];
```

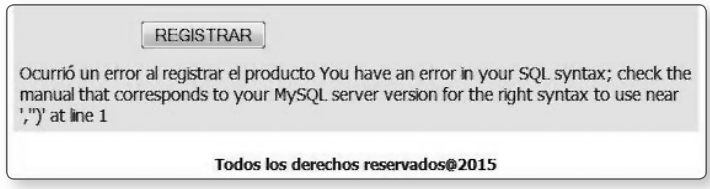
```
                $descripcion = $_POST['txtDescripcion'];
                $precio = $_POST['txtPrecio'];
                $stock = $_POST['txtStock'];
                $fecha = $_POST['txtFecha'];

                //Conectando al servidor y seleccionando la BD
                $cn=mysql_connect('localhost','root','');
                mysql_select_db('Ventas2015',$cn);

                //Sentencia de insercion de registros
                $rs=mysql_query("INSERT INTO producto
                                VALUES('$codigo','$descripcion',
                                $precio,$stock,'$fecha')",$cn);
                if ($rs)
                  echo "Registro de producto correcto..!!";
                  else
                  echo "Ocurrio un error al registrar el producto ".
                       mysql_error();
              }
            ?>
            </td>
          </tr>
          </table>
      </form>
      </section>
      <footer>
          <h5>Todos los derechos reservados@2015</h5>
      </footer>
  </body>
</html>
```

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
table {
    margin: auto;
    background-color: antiquewhite;
}
```

Caso 2: Implementar una aplicación en PHP que permita registrar los datos de los productos de forma básica usando el procedimiento almacenado para su registro.

En caso de que ocurra un error en el registro, se debe mostrar un mensaje al final de la siguiente forma:



En el caso de que el registro del producto sea correcto, entonces se debe mostrar el siguiente mensaje:

Procedimiento almacenado

```
DROP PROCEDURE IF EXISTS sp_insertaproducto;

CREATE PROCEDURE sp_insertaproducto(cod char(5),des varchar(50),
                                    pre decimal(8,2),sto int,
                                    fec date)
    INSERT INTO producto VALUES(cod,des,pre,sto,fec);
```

Archivo: **registro2.php**

```
<!DOCTYPE html>
<html>
 <head>
   <meta charset="UTF-8">
   <title></title>
   <link href="miEstilo.css" rel="stylesheet">
 </head>
 <body>
   <header>
```

```
  <h3>REGISTRO DE PRODUCTOS</h3>
  <h5>Usando procedimientos almacenados</h5>
</header>
<section>
  <form method="POST" action="registro.php">
    <table border="0" width="550"
           cellspacing="0" cellpadding="5">
      <tr>
        <td>CODIGO</td>
        <td><input type="text" name="txtCodigo"
                   value="" /></td>
</tr>
<tr>
  <td>DESCRIPCION</td>
  <td><input type="text" name="txtDescripcion"
             size="70" value="" /></td>
</tr>
<tr>
  <td>PRECIO</td>
  <td><input type="text" name="txtPrecio"
             value="" /></td>
</tr>
<tr>
  <td>STOCK</td>
  <td><input type="text" name="txtStock"
             value="" /></td>
</tr>
<tr>
  <td>FECHA DE VENC.</td>
  <td><input type="text" name="txtFecha"
             value="" /></td>
</tr>
<tr>
  <td></td>
  <td><input type="submit" name="btnRegistrar"
             value="REGISTRAR" /></td>
</tr>
<tr>
  <td colspan="2">
   <?php
     if (isset($_POST['btnRegistrar'])){
       //Capturando datos
       $codigo = $_POST['txtCodigo'];
       $descripcion = $_POST['txtDescripcion'];
       $precio = $_POST['txtPrecio'];
       $stock = $_POST['txtStock'];
       $fecha = $_POST['txtFecha'];

       //Conectando al servidor y seleccionando la BD
       $cn=mysql_connect('localhost','root','');
       mysql_select_db('Ventas2015',$cn);

       //Sentencia de insercion de registros
       $rs=mysql_query("CALL sp_insertaproducto('$codigo',
                        '$descripcion',
                        $precio,$stock,'$fecha')",$cn);
       if ($rs)
         echo "Registro de producto correcto..!!";
         else
         echo "Ocurrio un error al registrar el producto "
              .  mysql_error();
```

```
            }
          ?>
        </td>
      </tr>
    </table>
</form>
    </section>
    <footer>
 <h5>Todos los derechos reservados@2015</h5>
    </footer>
</body>
</html>
```

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
table {
    margin: auto;
    background-color: antiquewhite;
}
```

3.1.1 Casos desarrollados

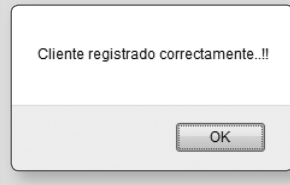
Caso desarrollado 1 **Registro del nuevo cliente**

Implementar una aplicación web con PHP y MySQL que permita registrar los datos de un nuevo cliente en la base de datos Ventas2015, tal como se muestra en la siguiente imagen:

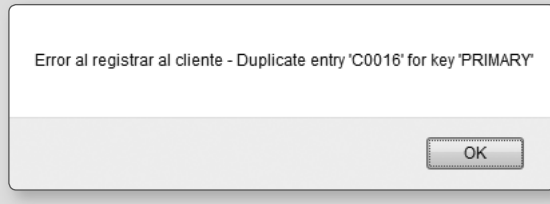
Imagen: registro.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **RegistroCliente.**
- Implementar procedimientos almacenados en MySQL que permitan registrar y listar los clientes de la base de datos Ventas2015.
- Los datos a registrar deben ser ingresados por el usuario inclusive el código del nuevo cliente, a excepción del distrito que deberá seleccionarse desde un cuadro combinado.
- Los distritos deben encontrarse precargados en el control cuadro combinado obtenidos desde la tabla Distrito de la base de datos Ventas2015.
- Al presionar el botón **REGISTRAR** debe registrar todos los datos en la tabla correspondiente, para lo cual debe mostrar un mensaje usando la función **ALERT** de Javascript de la siguiente manera:



- En caso ocurra un error en los datos registrados, se debe mostrar el siguiente mensaje usando la función **ALERT** de Javascript:



- En el pie de la página web se muestran enlaces para el listado de registros, así como para el nuevo registro. Respecto al listado de los clientes registrados, se debe mostrar tal como se presenta a continuación:

LISTADO DE CLIENTES

CÓDIGO	CLIENTE	DIRECCIÓN	TELÉFONO	DISTRITO	IMAGEN
C0001	CARLOS LOPEZ HURTADO	Calle El Pino 346	4677352	CERCADO	
C0002	FRIDA QUIROZ DUARTE	Jr. Los Reyes 475	6734732	CARABAYLLO	
C0003	ALEJANDRO TRAUCO MANRIQUE	Jr.Huascar 956	4676732	BARRANCO	

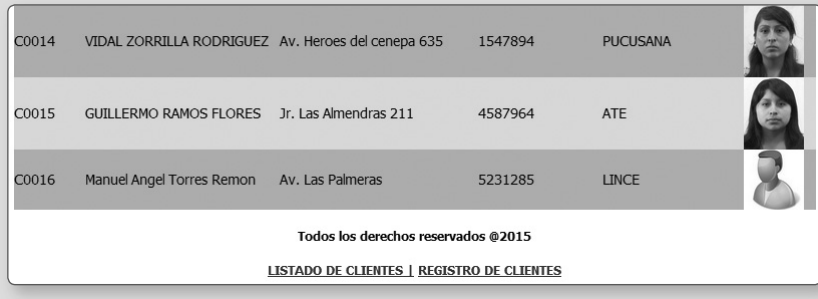


Imagen: listado.php

Nótese que al no registrar una imagen del cliente, se debe mostrar una imagen genérica como la siguiente: ; es decir, para este caso no se debe registrar la imagen del cliente, ya que automáticamente se asignará dicha imagen.

El objetivo de cada archivo del proyecto es:

- **conecta.php:** Contiene las sentencias que permite conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **registro.php:** Contiene la página que permite registrar los datos del nuevo cliente.
- **captura.php:** Contiene las funciones de captura de valor ingresadas en los controles del formulario.
- **listado.php:** Contiene la página que permite listar a los clientes registrados en la base de datos. Además de comprobar el nuevo registro de cliente, se debe tener en cuenta que los nuevos clientes deben mostrar una imagen genérica en la fotografía del cliente.

Para ello, se debe buscar una imagen parecida al ejemplo, llamarla sinfoto.jpg y colocarla dentro de la carpeta fotos del proyecto.

- **graba.php:** Contiene el código PHP que permite registrar los valores en la tabla Cliente, además de mostrar un mensaje según la acción emitida por la sentencia MySQL.
- **miEstilo.css:** Contiene los estilos que se aplicarán en el archivo **registro** y **listado.**

Procedimientos almacenados

```
DROP PROCEDURE IF EXISTS sp_distritos;
CREATE PROCEDURE sp_distritos()
    SELECT *
            FROM DISTRITO;

DROP PROCEDURE IF EXISTS sp_listadoclientes;
CREATE PROCEDURE sp_listadoclientes()
    SELECT C.ID_CLIENTE,
```

```
            CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
            C.DIRECCION,C.FONO,D.DESCRIPCION,C.EMAIL
            FROM CLIENTE C
            INNER JOIN DISTRITO D
            ON C.ID_DISTRITO=D.ID_DISTRITO
            ORDER BY 1 ASC;

DROP PROCEDURE IF EXISTS sp_nuevocliente;
CREATE PROCEDURE sp_nuevocliente(    COD CHAR(5),NOM VARCHAR(30),
                                     PAT VARCHAR(30),MAT VARCHAR(30),
                                     DIR VARCHAR(40),TEL CHAR(15),
                                     DIS CHAR(3),COR VARCHAR(50))
    INSERT INTO cliente VALUES(cod,nom,pat,mat,dir,tel,dis,cor);
```

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **registro.php**

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title></title>
    <link href="estilos.css" rel="stylesheet">
</head>
<body>
    <header>
        <h3>REGISTRO DEL NUEVO CLIENTE</h3>
        <img src="banner_clientes.jpg" width="700" height="250"/>
    </header>
    <section>
     <?php
        include('conexion.php');
     ?>
       <form method="POST" action="graba.php">
        <table border="0" width="700" cellspacing="0" cellpadding="0">
            <tr>
                <td width="500"></td>
                <td>CODIGO
                    <input type="text" name="txtCodigo"
                           maxlength="5" size="45"/>
                </td>
            </tr>
            <tr>
                <td colspan="2">NOMBRES</td>
            </tr>
            <tr>
                <td colspan="2">
                    <input type="text" name="txtNombres" size="112"/>
                </td>
            </tr>
            <tr>
                <td>PATERNO</td>
                <td>MATERNO</td>
```

```
            </tr>
            <tr>
                <td><input type="text" name="txtPaterno"
                           size="45"/>
                </td>
                <td><input type="text" name="txtMaterno"
                           size="45"/>
                </td>
            </tr>
            <tr>
                <td colspan="2">DIRECCION</td>
            </tr>
            <tr>
                <td colspan="2">
                    <input type="text" name="txtDireccion"
                           size="112"/>
                </td>
            </tr>
            <tr>
                <td>FONO</td>
                <td>DISTRITO</td>
            </tr>
            <tr>
                <td><input type="text" name="txtFono"
                           size="45"/>
                </td>
                <td><select name="selDistrito">
                      <?php
                         $rs=mysql_query('CALL SP_DISTRITOS',$cn);
                         while($distritos = mysql_fetch_array($rs)){
                           echo "<option value='".$distritos[0]."'>"
                                          .$distritos[1]."</option>";
                        }
                       ?>
                    </select></td>
            </tr>
            <tr>
                <td>CORREO ELECTRONICO</td>
                <td></td>
            </tr>
            <tr>
                <td colspan="2"><input type="text" name="txtCorreo"
                                       size="112"/></td>
            </tr>
            <tr>
                <td><input type="submit"
                           value="REGISTRAR" name="btnRegistrar" /></td>
                <td></td>
            </tr>
        </table>
    </form>
    </section>
    <footer>
        <h5>Todos los derechos reservados @2015</h5>
        <h5>
            <a href="listado.php">LISTADO DE CLIENTES |</a>
            <a href="registro.php">REGISTRO DE CLIENTES </a>
        </h5>
    </footer>
</body>
</html>
```

Comentarios

```
<?php
   include('conexion.php');
?>
```

Se incluye dentro del archivo **registro.php** las variables de conexión a la base de datos Ventas2015. Desde este momento se pueden usar dichas variables en el proceso de inserción de registro.

```
<?php
   $rs=mysql_query('CALL SP_DISTRITOS',$cn);
   $n=mysql_num_rows($rs);
   while($distritos = mysql_fetch_array($rs)){
      echo "<option value='".$distritos[0]."'>"
                              .$distritos[1]."</option>";
   }
?>
```

La variable **$rs** permite obtener los registros a partir de la consulta realizada al procedimiento almacenado **sp_distritos**. Ello se realiza para llenar el cuadro combinado con los nombres de los distritos.

Finalmente, se recorre por los distritos mediante la estructura **while**, lo que permitirá llenar el control cuadro combinado mediante la propiedad <OPTION>. Considere que **$distritos[0]** hace referencia al código del distrito; mientras que **$distritos[1]** hace referencia al nombre del distrito, el cual debe ser visualizado por el usuario.

Archivo: **captura.php**

```
<?php
    function getCodigo(){
        return $_POST['txtCodigo'];
    }
    function getNombres(){
        return $_POST['txtNombres'];
    }
    function getPaterno(){
        return $_POST['txtPaterno'];
    }
    function getMaterno(){
        return $_POST['txtMaterno'];
    }
    function getDireccion(){
        return $_POST['txtDireccion'];
    }
    function getTelefono(){
        return $_POST['txtFono'];
    }
    function getDistrito(){
        return $_POST['selDistrito'];
    }
    function getCorreo(){
        return $_POST['txtCorreo'];
    }
?>
```

Archivo: **graba.php**

```
<?php
    include('conexion.php');
    include('captura.php');

    $cod = getCodigo();
    $nom = getNombres();
    $pat = getPaterno();
    $mat = getMaterno();
    $dir = getDireccion();
    $tel = getTelefono();
    $dis = getDistrito();
    $cor = getCorreo();

    //Implementación de la sentencia de inserción del nuevo cliente
    $sql="CALL sp_nuevocliente('$cod','$nom','$pat','$mat',
                               '$dir','$tel','$dis','$cor')";

    //Ejecución de la sentencia
    $rpta=mysql_query($sql,$cn);
        if ($rpta){
            echo "<script>
                    alert ('Cliente registrado correctamente..!!');
                    history.back();
                  </script>";
        } else{
            echo '<script>
                alert ("Error al registrar al cliente-'.mysql_error().'");
                   history.back();
                  </script>';
        }
        mysql_close($cn);
?>
```

Comentarios

```
include('conexion.php');
include('captura.php');

$cod = getCodigo();
$nom = getNombres();
$pat = getPaterno();
$mat = getMaterno();
$dir = getDireccion();
$tel = getTelefono();
$dis = getDistrito();
$cor = getCorreo();

//Implementación de la sentencia de inserción del nuevo cliente
$sql="CALL sp_nuevocliente('$cod','$nom','$pat','$mat',
                           '$dir','$tel','$dis','$cor')";
```

Se incluyen los archivos **conexión.php** y **captura.php** para poder tener acceso a las variables de conexión y a los métodos que capturan los valores ingresados por el usuario, respectivamente.

Asimismo, se capturan todos los valores ingresados por el usuario a variables locales, los cuales servirán como valores de grabación al invocar el procedimiento almacenado que registrará al nuevo cliente. Finalmente, la variable **$sql** almacena la sentencia SQL que permite registrar un

nuevo cliente mediante la llamada al procedimiento almacenado **sp_nuevocliente**. Se debe tener en cuenta que los parámetros deben ser exactamente los especificados en el procedimiento almacenado y que los valores de tipo cadena deben registrarse con comillas simples, mientras que los numéricos se especifican directamente.

```
$rpta=mysql_query($sql,$cn);
if ($rpta){
   echo "<script>
            alert ('Cliente registrado correctamente..!!');
            history.back();
         </script>";
   } else{
     echo '<script>
            alert ("Error al registrar al cliente-'.mysql_error().'");
            history.back();
           </script>';
   }
mysql_close($cn);
```

La variable **$sql** debe ser ejecutada con la sentencia **mysql_query** para poder registrar de manera permanente en la base de datos. Se debe considerar que el resultado positivo o negativo de dicha función se captura en la variable **$rpta**, para luego enviar un mensaje de «Cliente registrado correctamente» o «Error al registrar al cliente» según sea el caso. Asimismo, considere que para enviar un mensaje de tipo alert, debe agregar las etiquetas <script> en la función **echo** de PHP.

Finalmente, se ha usado la función **history.back()** de Javascript para volver a la página origen después de mostrar los respectivos mensajes.

Archivo: **listado.php**

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title></title>
    <link href="estilos.css" rel="stylesheet">
</head>
<body>
    <?php
        include('conexion.php');
    ?>
  <header>
    <h3>LISTADO DE CLIENTES</h3>
    <img src="banner_clientes.jpg" width="800" height="250">
  </header>
  <section>
    <table id="tColor" width="800" border="0"
           cellpadding="0" cellspacing="0">
     <tr>
            <th width="80">CODIGO</th>
            <th width="280">CLIENTE</th>
            <th width="280">DIRECCION</th>
            <th width="160">TELEFONO</th>
```

```
            <th width="160">DISTRITO</th>
            <th width="100">IMAGEN</th>
    </tr>
      <?php
            $rs=mysql_query("CALL SP_LISTADOCLIENTES",$cn);
            while($clientes = mysql_fetch_array($rs)){
       ?>
    <tr>
            <td><?php echo $clientes[0];?></td>
            <td><?php echo $clientes[1];?></td>
            <td><?php echo $clientes[2];?></td>
            <td><?php echo $clientes[3];?></td>
            <td><?php echo $clientes[4];?></td>

            <td>
            <?php
                //Declarando el nombre del archivo a buscar
                $archivo = "fotos/".$clientes[0].".jpg";

                //Buscando el archivo
                if (file_exists($archivo))
                    echo "<img src=fotos/".$clientes[0].".jpg width=60>";
                else
                    echo "<img src=fotos/sinfoto.jpg width=60>";
            ?>
            </td>
    </tr>
        <?php
            }
       ?>
    </table>
  </section>
  <footer>
     <h5>Todos los derechos reservados @2015</h5>
     <h5>
        <a href="listado.php">LISTADO DE CLIENTES |</a>
        <a href="registro.php">REGISTRO DE CLIENTES </a>
     </h5>
  </footer>
</body>
</html>
```

Comentarios

```
<?php
   include('conexion.php');
?>
```

Se incluye dentro del archivo **listado.php** las variables de conexión que permiten conectarse a la base de datos Ventas2015.

```
<?php
   $rs=mysql_query("CALL SP_LISTADOCLIENTES",$cn);
   while($clientes = mysql_fetch_array($rs)){
?>
```

Se implementa la variable **$rs** que permite obtener los registros a partir de la ejecución del procedimiento almacenado SP_LISTADOCLIENTES desde la función **mysql_query**, no se debe olvidar especificar a qué conexión se refiere el procedimiento almacenado.

Para realizar la impresión de los registros obtenidos en la variable **$rs**, se debe recorrerlos por medio de una estructura repetitiva (**while**), de manera que se cree un arreglo de elementos llamado **$clientes** que contiene todos los registros de los clientes consultados, esto se debe a que la función **mysql_fetch_array** envía dicha información uno a uno al arreglo **$clientes**.

```
<tr>
   <td><?php echo $clientes[0];?></td>
   <td><?php echo $clientes[1];?></td>
   <td><?php echo $clientes[2];?></td>
   <td><?php echo $clientes[3];?></td>
   <td><?php echo $clientes[4];?></td>
   <td>
      <?php
         $archivo = "fotos/".$clientes[0].".jpg";

         if (file_exists($archivo))
            echo "<img src=fotos/".$clientes[0].".jpg width=60>";
            else
            echo "<img src=fotos/sinfoto.jpg width=60>";
      ?>
    </td>
</tr>
```

De acuerdo a lo especificado en el párrafo anterior, la variable **$clientes** contiene todos los registros de la consulta; ahora se deben descargar en sus respectivas celdas, de forma que al invocar a **$clientes[0]** se haga referencia al código del cliente. Solo en el caso de las imágenes se debe considerar que estas se encuentran en la carpeta **Fotos** y se debe comprobar su existencia; es decir, si se registra a un cliente nuevo, este no tendrá imagen, para comprobar ello se usa la sentencia **if(file_exists($archivo))**, donde **$archivo** contiene la dirección exacta de la supuesta imagen. Recuerde que la imagen de los clientes está compuesta por el código del mismo **($clientes[0])** y la extensión JPG.

Por otra parte, si la validación de la existencia del archivo es negativa, entonces se muestra la imagen **sinfoto.jpg** como imagen estándar para todos los clientes. Se debe considerar que la imagen **sinfoto.jpg** debe encontrarse previamente en la carpeta Fotos junto a las demás imágenes de los clientes.

Archivo: **estilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
img{
    margin: auto;
    display: block;
}
table{
    margin: auto;
}
```

```
/* Color para las filas impares */
#tColor tr:nth-child(odd) td {
    background-color: burlywood;
}

/* Color para las filas pares */
#tColor tr:nth-child(even) td {
    background-color: wheat;
 }
```

Caso desarrollado 2 **Registro del nuevo cliente adjuntando su imagen**

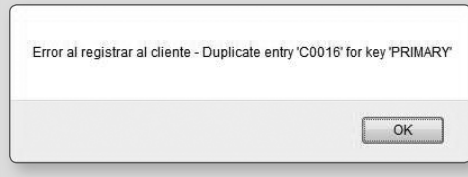
Implementar una aplicación web con PHP y MySQL que permita registrar los datos de un nuevo cliente en la base de datos Ventas2015 y en la que se permita incluir la imagen del cliente, tal como se muestra en la siguiente imagen:

Tener en cuenta lo siguiente:

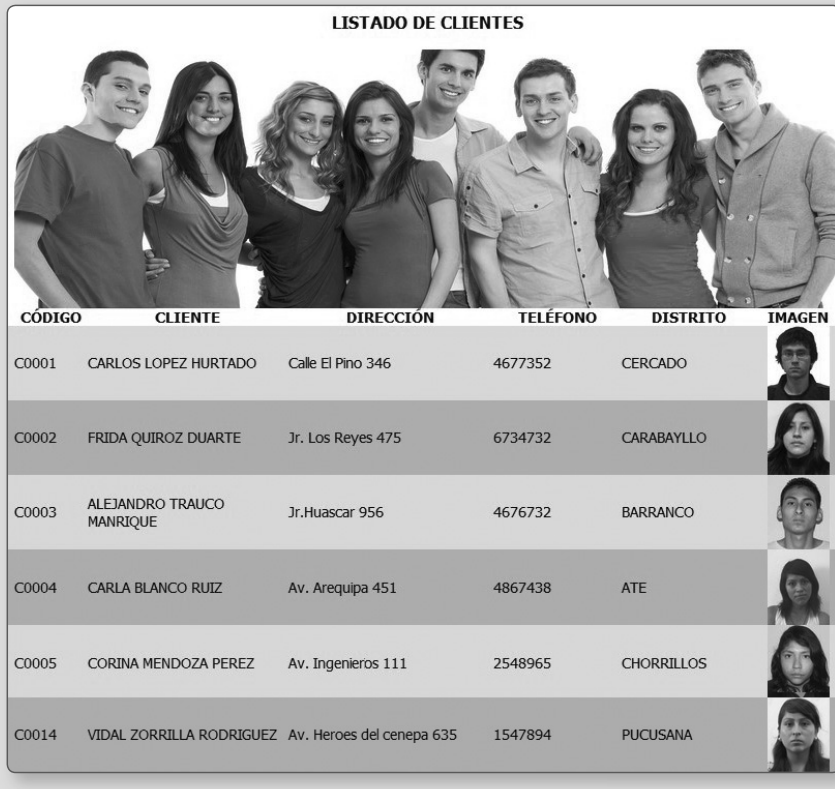
- Implementar toda la aplicación en una carpeta llamada **RegistroClienteFoto**.
- Implementar procedimientos almacenados tanto para el registro de clientes así como para el listado de los mismos.
- Los datos a registrar deben ser ingresados por el usuario a excepción del código del cliente que debe autogenerarse automáticamente y el distrito que deberá seleccionarse desde un cuadro combinado.

- Los distritos deben encontrarse precargados en el control cuadro combinado obtenidos desde la tabla Distrito.
- Para seleccionar la foto del cliente, se debe implementar el botón **Examinar** desde el control **File Select**.
- Si los datos ingresados son correctos, se debe grabar en la tabla mediante la opción REGISTRAR. Si todo es correcto, se debe mostrar el siguiente mensaje usando la función ALERT de Javascript:

- En caso ocurra algún error en los datos registrados, se debe mostrar el siguiente mensaje indicando el tipo de error, usando la función ALERT de Javascript:



- En el pie de página se deben mostrar los enlaces para listar y registrar los clientes. El listado mostrará todos los registros incluyendo sus imágenes:



LISTADO DE CLIENTES

CÓDIGO	CLIENTE	DIRECCIÓN	TELÉFONO	DISTRITO	IMAGEN
C0001	CARLOS LOPEZ HURTADO	Calle El Pino 346	4677352	CERCADO	
C0002	FRIDA QUIROZ DUARTE	Jr. Los Reyes 475	6734732	CARABAYLLO	
C0003	ALEJANDRO TRAUCO MANRIQUE	Jr.Huascar 956	4676732	BARRANCO	
C0004	CARLA BLANCO RUIZ	Av. Arequipa 451	4867438	ATE	
C0005	CORINA MENDOZA PEREZ	Av. Ingenieros 111	2548965	CHORRILLOS	
C0014	VIDAL ZORRILLA RODRIGUEZ	Av. Heroes del cenepa 635	1547894	PUCUSANA	

El objetivo de cada archivo del proyecto es:

- **conecta.php:** Contiene las sentencias que permiten conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **registro.php:** Contiene la página que permite registrar los datos del nuevo cliente incluyendo su foto.
- **captura.php:** Contiene las funciones de captura de todos los valores registrados en el formulario y autogenera el código del nuevo cliente en base al número de clientes registrados.
- **graba.php:** Contiene el código PHP que permite registrar los valores en la tabla Cliente. Tener en cuenta que si registra una imagen, se debe precargar con el botón **EXAMINAR** y el nombre del archivo de tipo JPG será el mismo del código autogenerado para el cliente.
- **listado.php:** Contiene la página que permite listar a los clientes para comprobar el nuevo registro de cliente. Tener en cuenta que si el nuevo cliente registró su imagen, este se listará; en caso contrario, se debe mostrar la imagen estándar **sinfoto.jpg**.
- **miEstilo.css:** Contiene los estilos que se aplicarán en el archivo **listado.php**.

Procedimientos almacenados

```
DROP PROCEDURE IF EXISTS sp_distritos;
CREATE PROCEDURE sp_distritos()
    SELECT *
            FROM DISTRITO;

DROP PROCEDURE IF EXISTS sp_listadoclientes;
CREATE PROCEDURE sp_listadoclientes()
    SELECT C.ID_CLIENTE,
            CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
            C.DIRECCION,C.FONO,D.DESCRIPCION,C.EMAIL
            FROM CLIENTE C
            INNER JOIN DISTRITO D
            ON C.ID_DISTRITO=D.ID_DISTRITO
            ORDER BY 1 ASC;

DROP PROCEDURE IF EXISTS sp_nuevocliente;
CREATE PROCEDURE sp_nuevocliente(    COD CHAR(5),NOM VARCHAR(30),
                                        PAT VARCHAR(30),MAT VARCHAR(30),
                                        DIR VARCHAR(40),TEL CHAR(15),
                                        DIS CHAR(3),COR VARCHAR(50))
    INSERT INTO cliente VALUES(cod,nom,pat,mat,dir,tel,dis,cor);
```

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **captura.php**

```
<?php
    function getCodigo(){
        include('conecta.php');
        $rs=mysql_query('SELECT * FROM CLIENTE');
        $n = mysql_num_rows($rs)+1;
        return 'C'. str_pad($n,4,'0',STR_PAD_LEFT);
    }
    function getNombres(){
        return $_POST['txtNombres'];
    }
    function getPaterno(){
        return $_POST['txtPaterno'];
    }
    function getMaterno(){
        return $_POST['txtMaterno'];
    }
    function getDireccion(){
        return $_POST['txtDireccion'];
    }
    function getTelefono(){
        return $_POST['txtFono'];
    }
    function getDistrito(){
        return $_POST['selDistrito'];
    }
    function getCorreo(){
        return $_POST['txtCorreo'];
    }
    function getFoto(){
        return $_FILES['filFoto'];
    }
?>
```

Comentarios

```
function getCodigo(){
  include('conecta.php');
  $rs=mysql_query('SELECT * FROM CLIENTE');
  $n = mysql_num_rows($rs)+1;
  return 'C'. str_pad($n,4,'0',STR_PAD_LEFT);
}
```

La función **getCodigo()** permite incluir la clase de conexión para conectarse con el servidor local y la base de datos Ventas2015, lo cual se hace para obtener el autogenerado de código. Para ello, se hace una consulta a la tabla Cliente para obtener el número de registros; por ejemplo, si se tiene 15 clientes, la variable **$n** almacenará dicho valor mediante la función **mysql_num_rows**, desde aquí se debe aumentar en un valor.

Finalmente, la función devolverá la concatenación de la letra inicial **C** y el número 0016. Para colocar los ceros a la izquierda se usa la función **str_pad**, cuyos parámetros son **$n** y contiene el número total de clientes más uno; el número 4, que representa el número de veces que se repetirá el valor siguiente; 0, que es el valor repetido, para este caso 4 veces, y, por último, la especificación que permite determinar para qué lado se insertarán los ceros, a la izquierda es STR_PAD_LEFT y a la derecha es STR_PAD_RIGTH.

Archivo: **registro.php**

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title></title>
    <link href="estilo.css" rel="stylesheet">
</head>
<body>
    <header>
        <h3>REGISTRO DEL NUEVO CLIENTE</h3>
        <img src="banner_clientes.jpg" width="700" height="250"/>
    </header>
    <section>
     <?php
        include('conecta.php');
        include('captura.php');
     ?>
        <form method="POST" action="graba.php" enctype="multipart/form-data">
         <table border="0" width="700" cellspacing="0" cellpadding="0">
            <tr>
                <td width="500"></td>
                <td>CODIGO
                    <input type="text" name="txtCodigo"
                           value="<?php echo getCodigo(); ?>"
                           maxlength="5" size="50" readonly />
                </td>
            </tr>
            <tr>
                <td colspan="2">NOMBRES</td>
            </tr>
            <tr>
                <td colspan="2">
                    <input type="text" name="txtNombres" size="112"/>
                </td>
            </tr>
            <tr>
                <td>PATERNO</td>
                <td>MATERNO</td>
            </tr>
            <tr>
                <td><input type="text" name="txtPaterno" size="50"/>
                </td>
                <td><input type="text" name="txtMaterno" size="50"/>
                </td>
            </tr>
            <tr>
                <td colspan="2">DIRECCION</td>
            </tr>
            <tr>
                <td colspan="2">
                    <input type="text" name="txtDireccion" size="112"/>
                </td>
            </tr>
            <tr>
                <td>FONO</td>
                <td>DISTRITO</td>
            </tr>
            <tr>
                <td><input type="text" name="txtFono" size="50"/>
                </td>
```

```
                <td><select name="selDistrito">
                      <?php
                         $rs=mysql_query('CALL SP_DISTRITOS',$cn);
                         $n=mysql_num_rows($rs);
                         while($distritos = mysql_fetch_array($rs)){
                           echo "<option value='".$distritos[0]."'>"
                                              .$distritos[1]."</option>";
                        }
                       ?>
                    </select></td>
            </tr>
            <tr>
                <td>CORREO ELECTRONICO</td>
                <td></td>
            </tr>
            <tr>
                <td colspan="2"><input type="text" name="txtCorreo"
                                       size="112"/></td>
            </tr>
            <tr>
                <td><input type="submit"
                           value="REGISTRAR" name="btnRegistrar" /></td>
                <td><input type="file" name="filFoto" value="" /></td>
            </tr>
        </table>
    </form>
  </section>
  <footer>
      <h5>Todos los derechos reservados @2015</h5>
      <h5>
          <a href="listado.php">LISTADO DE CLIENTES |</a>
          <a href="registro.php">REGISTRO DE CLIENTES </a>
      </h5>
  </footer>
</body>
</html>
```

Comentarios

```
<?php
  include('conecta.php');
  include('captura.php');
?>
```

Incluimos dentro del archivo **registro.php** las variables de conexión a la base de datos, desde el archivo **conecta.php** y los métodos que capturan los valores ingresados por el usuario por medio del archivo **captura.php**.

```
<form method="POST" action="graba.php" enctype="multipart/form-data">
        <table border="0" width="700" cellspacing="0" cellpadding="0">
```

Se debe tener en cuenta que en los casos anteriores los formularios no especificaban la acción a realizar después de presionar un SUBMIT. Para este caso, se debe especificar en la acción graba.php, que tiene la misión de registrar los datos del cliente en la tabla.

Finalmente, se debe especificar la propiedad enctype=”multipart/form-data” para el manejo correcto de las imágenes insertadas desde el formato HTML5.

```
<?php
  $rs=mysql_query('CALL SP_DISTRITOS',$cn);
  $n=mysql_num_rows($rs);
  while($distritos = mysql_fetch_array($rs)){
     echo "<option value='".$distritos[0]."'>"
                            .$distritos[1]."</option>";
  }
?>
```

Para llenar los nombres del distrito en el control cuadro combinado, primero se debe realizar la consulta a dicha tabla mediante el procedimiento almacenado SP_DISTRITOS. Dichos distritos serán enviados a la variable **$rs**, seguidamente se debe enviar dicha información, uno a uno, al arreglo **$distritos**. Desde aquí se enviará la información al control cuadro combinado mediante la propiedad <OPTION>. Finalmente, se debe tener en cuenta que **$distritos[0]** hace referencia a los códigos de distritos, mientras que **$distritos[1]**, al nombre del distrito.

Archivo: **graba.php**

```
<?php
    include('conecta.php');
    include('captura.php');

    //Capturando todos los valores del formulario
    $cod = getCodigo();
    $nom = getNombres();
    $pat = getPaterno();
    $mat = getMaterno();
    $dir = getDireccion();
    $tel = getTelefono();
    $dis = getDistrito();
    $cor = getCorreo();
    $fot = getFoto();

    //Subiendo la foto
    if (isset($fot)){
        $ruta ='fotos/';
        $nombre =  getCodigo().'.jpg';
        $foto = $_FILES['filFoto']['tmp_name'];
        $nombreFoto = $_FILES['filFoto']['name'];
        $subir = move_uploaded_file($foto, $ruta.$nombre);
    }

    //Implementación de la sentencia de inserción del nuevo cliente
    $sql="CALL sp_nuevocliente('$cod','$nom','$pat','$mat',
                               '$dir','$tel','$dis','$cor')";

    //Ejecución de la sentencia
    $rpta=mysql_query($sql,$cn);
        if ($rpta){
            echo "<script>
                    alert ('Cliente registrado correctamente..!!');
                    history.back();
                  </script>";
        } else{
            echo '<script>
                   alert ("Error al registrar al cliente-'.mysql_error().'");
                   history.back();
                  </script>';
        }
        mysql_close($cn);
?>
```

Comentarios

```
include('conecta.php');
include('captura.php');

$cod = getCodigo();
$nom = getNombres();
$pat = getPaterno();
$mat = getMaterno();
$dir = getDireccion();
$tel = getTelefono();
$dis = getDistrito();
$cor = getCorreo();
$fot = getFoto();
```

Se incluye la función **conecta.php**, que permite incluir las variables de conexión al servidor y acceso a la base de datos Ventas2015, y el archivo **captura.php**, que permite incluir las funciones de captura de valor desde los controles.

Desde aquí se puede descargar los valores capturados en variables, incluida la imagen seleccionada desde el control de tipo "file" obtenida desde la función **getFoto()**.

```
if (isset($fot)){
   $ruta ='fotos/';
   $nombre =  getCodigo().'.jpg';
   $foto = $_FILES['filFoto']['tmp_name'];
   $nombreFoto = $_FILES['filFoto']['name'];
   $subir = move_uploaded_file($foto, $ruta.$nombre);
}
```

Por otra parte, ahora se debe enviar la imagen seleccionada a la carpeta **Fotos** del proyecto. Se inicia verificando si la variable **$fot** contiene algún valor. Si contiene, se puede registrar el archivo, para lo cual se especifica la ruta de grabación en la variable **$ruta**, luego se define el nombre del archivo a partir del código capturado, además se agrega su extensión.

Asimismo, la variable **$foto** registra de manera temporal la imagen obtenida; de la misma manera, la variable **$subir** permite descargar la imagen en la carpeta destino, especificando en sus parámetros el nombre de la imagen registrada en el temporal (**$foto**) y la concatenación de la ruta y el nombre del archivo de tipo imagen.

```
 $sql="CALL sp_nuevocliente('$cod','$nom','$pat','$mat',
                            '$dir','$tel','$dis','$cor')";
```

La variable **$sql** registra la sentencia que invoca al procedimiento almacenado **sp_nuevocliente** con todos los parámetros del nuevo cliente.

Archivo: **listado.php**

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title></title>
    <link href="estilo.css" rel="stylesheet">
</head>
<body>
    <?php
        include('conecta.php');
    ?>
  <header>
    <h3>LISTADO DE CLIENTES</h3>
    <img src="banner_clientes.jpg" width="800" height="250">
  </header>
  <section>
    <table id="tColor" width="800" border="0"
           cellpadding="0" cellspacing="0">
    <tr>
            <th width="80">CODIGO</th>
            <th width="280">CLIENTE</th>
            <th width="280">DIRECCION</th>
            <th width="160">TELEFONO</th>
            <th width="160">DISTRITO</th>
            <th width="100">IMAGEN</th>
    </tr>
        <?php
            $rs=mysql_query("CALL SP_LISTADOCLIENTES",$cn);
            while($clientes = mysql_fetch_array($rs)){
      ?>
    <tr>
            <td><?php echo $clientes[0];?></td>
            <td><?php echo $clientes[1];?></td>
            <td><?php echo $clientes[2];?></td>
            <td><?php echo $clientes[3];?></td>
            <td><?php echo $clientes[4];?></td>

            <td>
            <?php
                //Declarando el nombre del archivo a buscar
                $archivo = "fotos/".$clientes[0].".jpg";

                //Buscando el archivo
                if (file_exists($archivo))
                    echo "<img src=fotos/".$clientes[0].".jpg width=60>";
                else
                    echo "<img src=fotos/sinfoto.jpg width=60>";
            ?>
            </td>
    </tr>
        <?php
            }
      ?>
    </table>
  </section>
  <footer>
     <h5>Todos los derechos reservados @2015</h5>
     <h5>
        <a href="listado.php">LISTADO DE CLIENTES |</a>
        <a href="registro.php">REGISTRO DE CLIENTES </a>
```

```
      </h5>
  </footer>
</body>
</html>
```

Comentarios

```
<?php
   include('conexion.php');
?>
```

Se incluye dentro del archivo **listado.php** las variables de conexión que permiten conectarse a la base de datos Ventas2015.

```
<?php
   $rs=mysql_query("CALL SP_LISTADOCLIENTES",$cn);
   while($clientes = mysql_fetch_array($rs)){
?>
```

Se implementa la variable **$rs** que permite obtener los registros a partir de la ejecución del procedimiento almacenado SP_LISTADOCLIENTES desde la función **mysql_query**. No se debe olvidar especificar a qué conexión se refiere el procedimiento almacenado.

Para realizar la impresión de los registros obtenidos en la variable **$rs**, se debe recorrerlos por medio de una estructura repetitiva (**while**), de manera que se cree un arreglo de elementos llamado **$clientes** que contiene todos los registros de los clientes consultados. Esto se debe a que la función **mysql_fetch_array** envía dicha información uno a uno al arreglo **$clientes**.

```
<tr>
   <td><?php echo $clientes[0];?></td>
   <td><?php echo $clientes[1];?></td>
   <td><?php echo $clientes[2];?></td>
   <td><?php echo $clientes[3];?></td>
   <td><?php echo $clientes[4];?></td>
   <td>
      <?php
         $archivo = "fotos/".$clientes[0].".jpg";

         if (file_exists($archivo))
            echo "<img src=fotos/".$clientes[0].".jpg width=60>";
            else
            echo "<img src=fotos/sinfoto.jpg width=60>";
      ?>
    </td>
</tr>
```

De acuerdo a lo especificado en el párrafo anterior, la variable **$clientes** contiene todos los registros de la consulta, ahora se debe descargarlos en sus respectivas celdas, de forma que al invocar a **$clientes[0]** se haga referencia al código del cliente. Solo en el caso de las imágenes se debe considerar que estas se encuentran en la carpeta **Fotos** y se debe comprobar su existencia, es decir, si se registra a un cliente nuevo, este no tendrá imagen. Para comprobarlo, se usa la sentencia **if(file_exists($archivo))**, donde **$archivo** contiene la dirección exacta de la supuesta imagen. Recuerde que la imagen de los clientes está compuesta por el código del mismo **($clientes[0])** y la extensión JPG.

Por otra parte, si la validación de la existencia del archivo es negativa, entonces se muestra la imagen **sinfoto.jpg**, como imagen estándar para todos los clientes. Se debe considerar que la imagen **sinfoto.jpg** debe encontrarse previamente en la carpeta **Fotos** junto a las demás imágenes de los clientes.

Archivo: **estilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
img{
    margin: auto;
    display: block;
}
table{
    margin: auto;
}

/* Color para las filas impares */
#tColor tr:nth-child(odd) td {
    background-color: burlywood;
}

/* Color para las filas pares */
#tColor tr:nth-child(even) td {
    background-color: wheat;
}
```

3.2 Actualización de registros

A. Comentario

La actualización de datos de un registro conlleva a la modificación o alteración de los valores de un registro determinado, se debe tener en cuenta que todos los valores se actualizarán dependiendo del campo clave. Para nuestro caso, se usará procedimientos almacenados de actualización.

B. Sintaxis

Sintaxis para la implementación del procedimiento almacenado que actualiza valores en una tabla de MySQL:

```
CREATE PROCEDURE sp_nombre_procedimiento(parametros)
  UPDATE tabla SET campo=nuevo_valor
  WHERE campo_clave=criterio;
```

Sintaxis para la llamada del procedimiento almacenado en PHP:

```
$variable = mysql_query("CALL sp_nombre_procedimiento(parametros)",$cn);
```

Sintaxis para la implementación de la sentencia de inserción sin procedimiento almacenado:

```
$variable = mysql_query("UPDATE tabla
                         SET campo=nuevo_valor
                         WHERE campo_clave=criterio",$cn);
```

Caso 1: Implementar una aplicación en PHP que permita actualizar los datos de los productos de forma básica usando la sentencia UPDATE directamente en la aplicación.

Primero, se debe mostrar los datos del producto mediante el botón **BUSCAR**, para ello se debe ingresar el código del producto; en caso no sea correcto, se debe mostrar el mensaje «Producto NO existe».

Una vez mostrado los datos en los controles, se podrá modificar los datos y grabarlos con el botón **ACTUALIZAR**. Si todo es correcto, se debe emitir el mensaje «Producto actualizado correctamente».

Archivo: **conecta.php**

```
<?php
    $cn=mysql_connect('localhost','root','')
                                        or die('Error en el Servidor');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **captura.php**

```
<?php
 function getDescripcion(){
return $_POST['txtDescripcion'];
 }
 function getPrecio(){
return $_POST['txtPrecio'];
 }
 function getStock(){
```

```
    return $_POST['txtStock'];
}
function getFecha(){
    return $_POST['txtFecha'];
 }
?>
```

Archivo: **actualiza.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta content="text/html" charset="ISO-8859-1">
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">
  </head>
  <body>
    <header>
      <?php include('encabezado.php'); ?>
      <h3>ACTUALIZACION DE DATOS DEL PRODUCTO</h3>
    </header>
    <section>
      <?php
        error_reporting(0);
        include 'conecta.php';
        include 'captura.php';
        $codigo = '';
        $mensaje='';

        if (isset($_POST['btnMantenimiento'])){
           $boton=$_POST['btnMantenimiento'];
           $codigo = $_POST['txtCodigo'];
           if ($boton=="BUSCAR"){
              $rs=  mysql_query("SELECT * FROM producto
                                 WHERE ID_PRODUCTO='$codigo'",$cn);

              $n=  mysql_num_rows($rs);
              if ($n==0)
                echo "<script>alert('Producto NO existe')</script>";

              $miProducto = mysql_fetch_array($rs);
            }

            if($boton=="ACTUALIZAR"){
              $cod= $codigo;
              $des= getDescripcion();
              $pre= getPrecio();
              $sto= getStock();
              $fec= getFecha();

              $rs=  mysql_query("UPDATE PRODUCTO
                                 SET DESCRIPCION='$des', PRECIO=$pre,
                                 STOCK=$sto, FECHA_VENC='$fec'
                                 WHERE ID_PRODUCTO='$cod'",$cn);
        $n=  mysql_num_rows($rs);
        if ($n!=0)
          echo "<script>
                  alert('Producto actualizado correctamente')
                </script>";
        }
    }
 ?>
 <form method="POST">
```

```
    <table border="1" width="550"
           cellspacing="0" cellpadding="10">
      <tr>
         <td>Ingrese codigo de producto</td>
         <td><input type="text" name="txtCodigo"
                    value="<?php echo $miProducto[0]; ?>" /></td>
         <td><input type="submit" name="btnMantenimiento"
                    value="BUSCAR"/><?php echo $mensaje; ?></td>
      </tr>
    </table>

    <table border="1" width="550"
           cellspacing="0" cellpadding="10">
      <tr>
         <td>DESCRIPCION</td>
         <td colspan="3">
             <input type="text" name="txtDescripcion" size="62"
                    value="<?php echo $miProducto[1]; ?>" />
         </td>
      </tr>
      <tr>
         <td>PRECIO</td>
         <td><input type="text" name="txtPrecio"
                   value="<?php echo $miProducto[2]; ?>" /></td>
         <td>STOCK</td>
         <td><input type="text" name="txtStock"
                   value="<?php echo $miProducto[3]; ?>" /></td>
      </tr>
      <tr>
         <td>FECHA VENC.</td>
         <td><input type="text" name="txtFecha"
                    value="<?php echo $miProducto[4]; ?>" /></td>
         <td colspan="2">
                <input type="submit" name="btnMantenimiento"
                                     value="ACTUALIZAR" /></td>
      </tr>
  </table>
</form>
</section>
<footer></footer>
 </body>
</html>
```

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
table{
    margin: auto;
}
img{
    margin: auto;
    display: block;
}
h3,h5{
    text-align: center;
}
#centrado{
    text-align: center;
}
td {
```

```
    border: solid 1px #006699;
    border-top: #006699;
    border-right: #006699;
    border-left: #006699;
    border-bottom: #006699;
}
```

Caso 2: Implementar una aplicación en PHP que permita actualizar los datos de los productos de forma básica usando la sentencia UPDATE mediante un procedimiento almacenado.

Primero, se debe mostrar los datos del producto mediante el botón **BUSCAR**, para ello debe ingresar el código del producto, en caso no sea correcto se debe mostrar el mensaje «Producto NO existe».

Una vez mostrado los datos en los controles, se podrá modificar los datos y grabarlos con el botón **ACTUALIZAR**; si todo es correcto, se debe emitir el mensaje «Producto actualizado correctamente».

Implemente procedimientos almacenados que permitan buscar un determinado producto y otro para la actualización de los datos del producto.

Procedimientos almacenados

```
DROP PROCEDURE IF EXISTS sp_buscaProducto;
CREATE PROCEDURE sp_buscaProducto(cod char(5))
    SELECT * FROM producto
            WHERE ID_PRODUCTO=cod;

DROP PROCEDURE IF EXISTS sp_actualizaProducto;
CREATE PROCEDURE sp_actualizaProducto(cod char(5),des varchar(50),
                                pre decimal(8,2),sto int, fec date)
    UPDATE PRODUCTO
        SET DESCRIPCION=des, PRECIO=pre, STOCK=sto, FECHA_VENC=fec
        WHERE ID_PRODUCTO=cod;
```

Archivo: **conecta.php**

```
<?php
    $cn=mysql_connect('localhost','root','')
                                        or die('Error en el Servidor');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **captura.php**

```
<?php
 function getDescripcion(){
return $_POST['txtDescripcion'];
 }
 function getPrecio(){
return $_POST['txtPrecio'];
 }
 function getStock(){
return $_POST['txtStock'];
 }
 function getFecha(){
return $_POST['txtFecha'];
 }
?>
```

Archivo: **actualiza.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta content="text/html" charset="ISO-8859-1">
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">
  </head>
  <body>
    <header>
      <?php include('encabezado.php'); ?>
      <h3>ACTUALIZACION DE DATOS DEL PRODUCTO</h3>
    </header>
    <section>
      <?php
        error_reporting(0);
        include 'conecta.php';
        include 'captura.php';
        $codigo = '';
        $mensaje='';

        if (isset($_POST['btnMantenimiento'])){
           $boton=$_POST['btnMantenimiento'];
           $codigo = $_POST['txtCodigo'];
           if ($boton=="BUSCAR"){
              $rs=mysql_query("CALL sp_buscaProducto('$codigo')",$cn);

              $n=  mysql_num_rows($rs);
              if ($n==0)
                echo "<script>alert('Producto NO existe')</script>";

              $miProducto = mysql_fetch_array($rs);
            }

            if($boton=="ACTUALIZAR"){
```

```
                $cod = $codigo;
                $des = getDescripcion();
                $pre = getPrecio();
                $sto = getStock();
                $fec = getFecha();

                $rs = mysql_query("CALL sp_actualizaProducto('$cod',
                                        '$des',$pre,$sto,'$fec')",$cn);
          $n =  mysql_num_rows($rs);
          if ($n!=0)
            echo "<script>
                    alert('Producto actualizado correctamente')
                  </script>";
          }
    }
  ?>
  <form method="POST">
     <table border="1" width="550"
            cellspacing="0" cellpadding="10">
       <tr>
          <td>Ingrese codigo de producto</td>
          <td><input type="text" name="txtCodigo"
                    value="<?php echo $miProducto[0]; ?>" /></td>
          <td><input type="submit" name="btnMantenimiento"
                    value="BUSCAR"/><?php echo $mensaje; ?></td>
       </tr>
     </table>

     <table border="1" width="550"
            cellspacing="0" cellpadding="10">
       <tr>
          <td>DESCRIPCION</td>
          <td colspan="3">
              <input type="text" name="txtDescripcion" size="62"
                     value="<?php echo $miProducto[1]; ?>" />
          </td>
       </tr>
       <tr>
          <td>PRECIO</td>
          <td><input type="text" name="txtPrecio"
                    value="<?php echo $miProducto[2]; ?>" /></td>
          <td>STOCK</td>
          <td><input type="text" name="txtStock"
                    value="<?php echo $miProducto[3]; ?>" /></td>
       </tr>
       <tr>
          <td>FECHA VENC.</td>
          <td><input type="text" name="txtFecha"
                     value="<?php echo $miProducto[4]; ?>" /></td>
          <td colspan="2">
                 <input type="submit" name="btnMantenimiento"
                                      value="ACTUALIZAR" /></td>
       </tr>
    </table>
  </form>
  </section>
  <footer></footer>
   </body>
  </html>
```

Archivo: **miEstilo.css**

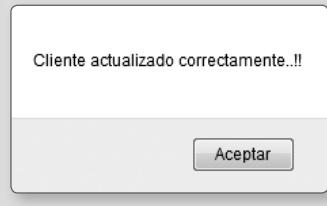
```
body{
    font-family: tahoma;
    font-size: 14px;
}
table{
    margin: auto;
}
img{
    margin: auto;
    display: block;
}
h3,h5{
    text-align: center;
}
#centrado{
    text-align: center;
}
td {
    border: solid 1px #006699;
    border-top: #006699;
    border-right: #006699;
    border-left: #006699;
    border-bottom: #006699;
}
```

Caso desarrollado **Actualización de datos del cliente**

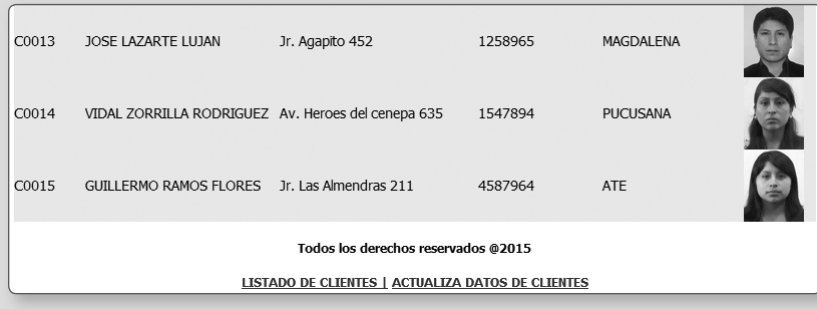
Implementar una aplicación web con PHP y MySQL que permita actualizar los datos de un cliente en la base de datos Ventas2015, tal como se muestra en la siguiente imagen:

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **ActualizaCliente**.
- Implementar procedimientos almacenados para listar, buscar y actualizar los datos de los clientes.
- Los distritos deben encontrarse precargados en el control cuadro combinado obtenidos desde la tabla Distrito.
- Antes de actualizar los datos, se debe buscar a un determinado cliente por su código; una vez encontrado se deberá mostrar sus datos en los controles respectivos, tal es así que al inicio el cuadro combinado de los distritos debe mostrar todos los distritos desde la tabla Distrito y al buscar a un cliente, en el cuadro se mostrará el distrito al que pertenece el cliente.
- Una vez mostrados los datos del cliente a modificar, podrá realizar las modificaciones que crea conveniente y registrarlos con el botón **MODIFICAR**.
- Si los datos actualizados son correctos, se deben registrar en la tabla mediante el botón **MODIFICAR**, además de mostrar el mensaje «Cliente actualizado correctamente..!!» usando la función ALERT de Javascript:



- En el pie de página se debe mostrar los enlaces para listar y actualizar los datos de los clientes, el listado mostrará todos los registros de los clientes, incluyendo sus imágenes:



El objetivo de cada archivo del proyecto es:

- **conecta.php:** Contiene las sentencias que permite conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **actualiza.php:** Contiene la página que permite buscar y actualizar los datos del cliente.
- **captura.php:** Contiene las funciones de captura de todos los valores registrados en el formulario.
- **listado.php:** Contiene la página que permite listar a los clientes para comprobar la actualización de los datos del cliente.
- **miEstilo.css:** Contiene los estilos que se aplicarán a los archivos **listado.php** y **actualiza.php**.

Procedimientos almacenados

```
DROP PROCEDURE IF EXISTS sp_listadoclientes;
CREATE PROCEDURE sp_listadoclientes()
    SELECT C.ID_CLIENTE,
           CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
           C.DIRECCION,C.FONO,D.DESCRIPCION,C.EMAIL
           FROM CLIENTE C
           INNER JOIN DISTRITO D
           ON C.ID_DISTRITO=D.ID_DISTRITO
           ORDER BY 1 ASC;

DROP PROCEDURE IF EXISTS sp_buscacliente;
CREATE PROCEDURE SP_BUSCACLIENTE(COD CHAR(5))
    SELECT * FROM CLIENTE WHERE ID_CLIENTE=COD;

DROP PROCEDURE IF EXISTS sp_actualizacliente;
CREATE PROCEDURE sp_actualizacliente(COD CHAR(5),NOM VARCHAR(30),
                                     PAT VARCHAR(30),MAT VARCHAR(30),
                                     DIR VARCHAR(40),TEL CHAR(15),
                                     DIS CHAR(3),COR VARCHAR(50))
    UPDATE CLIENTE
           SET NOMBRES=NOM, PATERNO=PAT, MATERNO=MAT, DIRECCION=DIR,
                  FONO=TEL, ID_DISTRITO=DIS, EMAIL=COR
           WHERE ID_CLIENTE=COD;
```

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **captura.php**

```
<?php
    function getCodigo(){
        return $_POST['txtCodigo'];
    }
    function getNombres(){
        return $_POST['txtNombres'];
    }
    function getPaterno(){
        return $_POST['txtPaterno'];
    }
    function getMaterno(){
        return $_POST['txtMaterno'];
    }
    function getDireccion(){
        return $_POST['txtDireccion'];
    }
    function getTelefono(){
        return $_POST['txtFono'];
    }
    function getDistrito(){
        return $_POST['selDistrito'];
    }
    function getCorreo(){
        return $_POST['txtCorreo'];
    }
?>
```

Archivo: **actualiza.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">
  </head>
  <body>
    <header>
      <h3>ACTUALIZACION DE DATOS DEL CLIENTE</h3>
      <img src="banner_clientes.jpg" width="550" height="180"/>
    </header>
    <section>
      <?php
        error_reporting(0);
        include('conecta.php');
        include('captura.php');

        //Consultando la lista de Distritos para llenar
        //en el cuadro combinado
        $rsD = mysql_query('SELECT * FROM DISTRITO', $cn);
        $codigo = getCodigo();

        //Determina que  ódig ha seleccionado el usuario
        if (isset($_POST['btnGenerico'])) {
           $boton = $_POST['btnGenerico'];

           if ($boton == 'BUSCAR') {
             $rs = mysql_query("CALL sp_buscacliente('$codigo');", $cn);
```

```
            $cliente = mysql_fetch_array($rs);
          }
          if ($boton == 'MODIFICAR'){
            $cod = getCodigo();
            $nom = getNombres();
            $pat = getPaterno();
            $mat = getMaterno();
            $dir = getDireccion();
            $tel = getTelefono();
            $dis = getDistrito();
            $cor = getCorreo();

            //Implementación de la sentencia de  ódigo zación del cliente
            $sql="CALL sp_actualizacliente('$cod','$nom','$pat','$mat',
                                          '$dir','$tel','$dis','$cor')";

            //Ejecución de la sentencia de  ódigo zación
            $rpta=mysql_query($sql,$cn);
            if ($rpta)
               echo "<script>
                        alert ('Cliente actualizado correctamente..!!');
                        history.back();
                     </script>";
              else
               echo '<script>
                 alert ("Error al actualizar al cliente-'.mysql_error().'");
                 history.back();
                 </script>';
          }
      }
?>
<form method="POST">
   <table border="0" width="550" cellspacing="1" cellpadding="10">
      <tr>
         <td>Ingrese código</td>
         <td><input type="text" name="txtCodigo" maxlength="5"
                    value="<?php echo getCodigo(); ?>" /></td>
         <td><input type="submit" name="btnGenerico"
                    value="BUSCAR" /></td>
      </tr>
    </table>
    <br>
    <table border="0" width="550" cellspacing="1" cellpadding="1">
       <tr>
         <td>NOMBRES</td>
         <td colspan="3">
           <input type="text" name="txtNombres"
                  size="70" value="<?php echo $cliente[1] ?>" /></td>
       </tr>
       <tr>
         <td>PATERNO</td>
         <td>
            <input type="text" name="txtPaterno"
                   value="<?php echo $cliente[2] ?>" />
         </td>
         <td>MATERNO</td>
         <td>
            <input type="text" name="txtMaterno"
                   value="<?php echo $cliente[3] ?>" />
         </td>
       </tr>
       <tr>
```

```
            <td>DIRECCION</td>
            <td colspan="3">
               <input type="text" name="txtDireccion"
                      size="70" value="<?php echo $cliente[4] ?>" />
            </td>
         </tr>
         <tr>
            <td>TELEFONO</td>
            <td>
               <input type="text" name="txtFono"
                      value="<?php echo $cliente[5] ?>" />
            </td>
            <td>DISTRITO</td>
            <td>
               <select name="selDistrito">
                  <?php
                     //Recorriendo por todos los distritos
                     while ($distritos = mysql_fetch_array($rsD)) {
                        if ($cliente[6] == $distritos[0])
                            $seleccionado='SELECTED';
                            else
                            $seleccionado='';
                  ¿>
                  <option  VALUE="<?php echo $distritos[0]; ¿>"
                           <?php echo $seleccionado; ¿>>
                           <?php echo $distritos[1] ¿>
                  </option>
                  <?php } ?>
               </select>
             </td>
         </tr>
         <tr>
            <td>EMAIL</td>
            <td colspan="3">
               <input type="text" name="txtCorreo"
                      size="70" value="<?php echo $cliente[7] ?>" /></td>
         </tr>
         <tr>
            <td colspan="4"><input type="submit"
                name="btnGenerico"  value="MODIFICAR" /></td>
         </tr>
        </table>
      </form>
    </section>
    <footer>
       <h5>Todos los derechos reservados @2015</h5>
       <h5>
         <a href="listado.php">LISTADO DE CLIENTES |</a>
         <a href="actualiza.php">ACTUALIZA DATOS DE CLIENTES </a>
       </h5>
    </footer>
</body>
</html>
```

Comentarios

```
<?php
   error_reporting(0);
   include('conecta.php');
   include('captura.php');

   $rsD = mysql_query('SELECT * FROM DISTRITO', $cn);
   $codigo = getCodigo();

   if (isset($_POST['btnGenerico'])) {
      $boton = $_POST['btnGenerico'];

      if ($boton == 'BUSCAR') {
        $rs = mysql_query("CALL sp_buscacliente('$codigo');", $cn);
        $cliente = mysql_fetch_array($rs);
      }
      if ($boton == 'MODIFICAR'){
        $cod = getCodigo();
        $nom = getNombres();
        $pat = getPaterno();
        $mat = getMaterno();
        $dir = getDireccion();
        $tel = getTelefono();
        $dis = getDistrito();
        $cor = getCorreo();

        $sql="CALL sp_actualizacliente('$cod','$nom','$pat','$mat',
                                       '$dir','$tel','$dis','$cor')";

        $rpta=mysql_query($sql,$cn);
        if ($rpta)
           echo "<script>
           alert ('Cliente actualizado correctamente..!!');
           history.back();
           </script>";
           else
             echo '<script>
                alert ("Error al actualizar al cliente-'.mysql_error().'");
                history.back();
             </script>';
           }
       }
¿>
```

Se inicia con la inclusión de los archivos **conecta.php** y **captura.php**. El primero incorpora las variables de conexión al servidor y acceso a la base de datos, mientras que el segundo incluye funciones de captura de valor, las cuales obtienen los datos registrados por el usuario al momento de actualizar sus datos.

Asimismo, se declara la variable **$rsD** que permite ejecutar la consulta de los distritos usando la función **mysql_query**. Seguidamente, se envía el código ingresado por el usuario a la variable **$codigo** para luego ser usado como parámetro de búsqueda.

Se debe tener en cuenta que todos los botones de la página tienen asignados el nombre **btnGenerico**, esto se realiza con el propósito de poder condicionar qué botón ha seleccionado el usuario, puesto que se sabe que un formulario solo puede realizar un *submit* por vez. En este sentido, se obtiene el nombre del botón presionado por el usuario mediante la variable **$boton;**

desde aquí se puede realizar una u otra acción dentro de la web, si el botón seleccionado es **BUSCAR**, entonces se implementa la variable **$rs** con la ejecución de la sentencia que invoca al procedimiento almacenado SP_BUSCACLIENTE, el cual solicita un código para la búsqueda del cliente; esto se encuentra registrado en la variable **$codigo**.

```
if (isset($_POST['btnGenerico'])) {
      $boton = $_POST['btnGenerico'];

      if ($boton == 'BUSCAR') {
        $rs = mysql_query("CALL SP_BUSCACLIENTE('$codigo');", $cn);
        $cliente = mysql_fetch_array($rs);
}
```

Por otra parte, si el botón seleccionado es **MODIFICAR**, entonces se obtienen todos los valores desde la función **get** a las variables **$cod**, **$nom**, **$pat**, **$mat**, **$dir**, **$tel**, **$dis**, **$cor**, respectivamente; y, finalmente, se envía en el procedimiento almacenado SP_ACTUALIZACLIENTE especificado mediante la variable **$sql**. Recuerde que por tratarse de columnas de tipo cadena o fecha se debe colocar los parámetros entre comillas simples.

```
<tr>
   <td>NOMBRES</td>
   <td colspan="3">
       <input type="text" name="txtNombres"
              size="70" value="<?php echo $cliente[1] ?>" /></td>
</tr>
```

Se imprime dentro de la celda correspondiente los valores encontrados en la variable de tipo arreglo **$cliente[1]** que pertenece al nombre del cliente; esto se debe a la consulta del tipo búsqueda.

```
<td>DISTRITO</td>
<td>
   <select name="selDistrito">
       <?php
          while ($distritos = mysql_fetch_array($rsD)) {
          if ($cliente[6] == $distritos[0])
             $seleccionado='SELECTED';
             else
             $seleccionado='';
       ?>
        <option  VALUE="<?php echo $distritos[0]; ?>"
                        <?php echo $seleccionado; ?>>
                        <?php echo $distritos[1] ?>
        </option>
       <?php } ?>
    </select>
</td>
```

Finalmente, se debe llenar el control de cuadro combinado con los distritos procedentes de la búsqueda del cliente. Considere que la búsqueda del cliente devuelve el código del distrito, por ejemplo, L04. A partir de aquí se debe buscar el nombre del distrito para mostrarlo en el cuadro combinado como valor activo.

Para ello, primero se debe recorrer por todos los distritos especificados en la variable **$rsD**, luego comparar el código encontrado ($cliente[6]) con el código del distrito ($distrito[0]) y asignar a la variable **$seleccionado** la opción SELECTED. Si no se encuentra dicha variable, será asignada como vacía, ya que podría ser que el distrito no se encuentre en la tabla Distrito.

Finalmente, todos los datos son enviados al control SELECT mediante la propiedad <OPTION>, el parámetro inicial especificado del valor estará asignado por cada distrito ($distritos[0]), seguidamente se especifica la sentencia SELECT para ubicarlo en un distrito determinado desde la impresión de la variable **$seleccionado**. Lo que el usuario debe observar se encuentra en **$distritos[1]** y hace referencia al nombre del distrito.

Archivo: **listado.php**

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">
</head>
<body>
    <?php
        include('conecta.php');
    ?>
  <header>
    <h3>LISTADO DE CLIENTES</h3>
    <img src="banner_clientes.jpg" width="800" height="250">
  </header>
  <section>
    <table width="800" border="0"
           cellpadding="0" cellspacing="0">
    <tr>
            <th width="80">CODIGO</th>
            <th width="280">CLIENTE</th>
            <th width="280">DIRECCION</th>
            <th width="160">TELEFONO</th>
            <th width="160">DISTRITO</th>
            <th width="100">IMAGEN</th>
    </tr>
      <?php
           $rs=mysql_query("CALL SP_LISTADOCLIENTES",$cn);
           while($clientes = mysql_fetch_array($rs)){
       ?>
    <tr>
            <td><?php echo $clientes[0];?></td>
            <td><?php echo $clientes[1];?></td>
            <td><?php echo $clientes[2];?></td>
            <td><?php echo $clientes[3];?></td>
            <td><?php echo $clientes[4];?></td>

            <td>
            <?php
                //Declarando el nombre del archivo a buscar
                $archivo = "fotos/".$clientes[0].".jpg";

                //Buscando el archivo
                if (file_exists($archivo))
                    echo "<img src=fotos/".$clientes[0].".jpg width=60>";
                else
```

```
                    echo "<img src=fotos/sinfoto.jpg width=60>";
                ?>
                </td>
        </tr>
          <?php
                }
           ?>
        </table>
    </section>
    <footer>
        <h5>Todos los derechos reservados @2015</h5>
        <h5>
            <a href="listado.php">LISTADO DE CLIENTES |</a>
            <a href="actualiza.php">ACTUALIZA DATOS DE CLIENTES </a>
        </h5>
    </footer>
</body>
</html>
```

Comentarios

```
<?php
    include('conexion.php');
?>
```

Se incluye dentro del archivo **listado.php** las variables de conexión que permiten conectarse a la base de datos Ventas2015.

```
<?php
    $rs=mysql_query("CALL SP_LISTADOCLIENTES",$cn);
    while($clientes = mysql_fetch_array($rs)){
?>
```

Se implementa la variable **$rs** que permite obtener los registros a partir de la ejecución del procedimiento almacenado SP_LISTADOCLIENTES desde la función **mysql_query**. No se debe olvidar especificar a qué conexión se refiere el procedimiento almacenado.

Para realizar la impresión de los registros obtenidos en la variable **$rs**, se debe recorrerlos por medio de una estructura repetitiva (**while**), de manera que se cree un arreglo de elementos llamado **$clientes**, el cual contiene todos los registros de los clientes consultados, esto se debe a que la función **mysql_fetch_array** envía dicha información, uno a uno, al arreglo **$clientes**.

```
<tr>
    <td><?php echo $clientes[0];?></td>
    <td><?php echo $clientes[1];?></td>
    <td><?php echo $clientes[2];?></td>
    <td><?php echo $clientes[3];?></td>
    <td><?php echo $clientes[4];?></td>
    <td>
        <?php
            $archivo = "fotos/".$clientes[0].".jpg";

            if (file_exists($archivo))
                echo "<img src=fotos/".$clientes[0].".jpg width=60>";
```

```
            else
            echo "<img src=fotos/sinfoto.jpg width=60>";
        ?>
      </td>
    </tr>
```

De acuerdo a lo especificado en el párrafo anterior, la variable **$clientes** contiene todos los registros de la consulta, ahora se deben descargar en sus respectivas celdas, de forma que al invocar a **$clientes[0]** se haga referencia al código del cliente. Solo en el caso de las imágenes se debe considerar que estas se encuentran en la carpeta **Fotos** y se debe comprobar su existencia, es decir, si se registra a un cliente nuevo este no tendrá imagen; para comprobarlo se usa la sentencia **if(file_exists($archivo))**, donde **$archivo** contiene la dirección exacta de la supuesta imagen. Recuerde que la imagen de los clientes está compuesta por el código del mismo **($clientes[0])** y la extensión JPG.

Por otra parte, si la validación de la existencia del archivo es negativa, entonces se debe mostrar la imagen **sinfoto.jpg** como imagen estándar para todos los clientes. Se debe considerar que la imagen **sinfoto.jpg** debe encontrarse previamente en la carpeta **Fotos** junto a las demás imágenes de los clientes.

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
#centrado{
    text-align: center;
}
table {
    margin: auto;
    background-color: antiquewhite;
}
img{
    margin: auto;
    display: block;
}
table th {
background-color: #b9c9fe;
}
```

3.3 Eliminación de registros

A. Comentario

La eliminación de registros desde una tabla conlleva una serie de restricciones que se deben considerar antes de ejecutar la aplicación, la principal restricción es que solo se pueden eliminar registros que no se encuentren asociados a otras tablas; esta utilidad de eliminar podría usarse cuando desea eliminar una orden de pedido o eliminar una línea de venta de pedido.

B. Sintaxis

Sintaxis para la implementación del procedimiento almacenado que elimina un registro en una tabla de MySQL:

```
CREATE PROCEDURE sp_nombre_procedimiento(parametros)
  DELETE FROM tabla WHERE campo=valor;
```

Sintaxis para la llamada del procedimiento almacenado en PHP:

```
$variable = mysql_query("CALL sp_nombre_procedimiento(parametros)",$cn);
```

Sintaxis para la implementación de la sentencia de inserción sin procedimiento almacenado:

```
$variable = mysql_query("DELETE FROM tabla WHERE campo=valor",$cn);
```

Caso 1: Implementar una aplicación en PHP que permita eliminar los datos de un determinado producto de forma básica usando la sentencia DELETE mediante un procedimiento almacenado.

Primero se debe mostrar los datos del producto mediante el botón **BUSCAR**, para esto deberá ingresar el código del producto; en caso no sea correcto, mostrar el mensaje «Producto NO existe».

Una vez mostrados los datos, en los controles se podrá eliminar los datos del producto con el botón **ELIMINAR**. Si todo es correcto, emitir el mensaje «Producto eliminado correctamente».

Implemente procedimientos almacenados que permitan buscar un producto y otro para la eliminación del mismo.

Procedimientos almacenados

```
DROP PROCEDURE IF EXISTS sp_buscaProducto;
CREATE PROCEDURE sp_buscaProducto(cod char(5))
    SELECT * FROM producto
            WHERE ID_PRODUCTO=cod;

DROP PROCEDURE IF EXISTS sp_eliminaProducto;
CREATE PROCEDURE sp_eliminaProducto(cod char(5))
    DELETE FROM PRODUCTO
            WHERE ID_PRODUCTO=cod;
```

Archivo: **conecta.php**

```
<?php
    $cn=mysql_connect('localhost','root','')
                                    or die('Error en el Servidor');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **captura.php**

```
<?php
 function getDescripcion(){
      return $_POST['txtDescripcion'];
 }
 function getPrecio(){
      return $_POST['txtPrecio'];
 }
 function getStock(){
      return $_POST['txtStock'];
 }
 function getFecha(){
      return $_POST['txtFecha'];
 }
?>
```

Archivo: **elimina.php**

```
<!DOCTYPE html>
<html>
 <head>
    <meta content="text/html" charset="ISO-8859-1">
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">
 </head>
 <body>
   <header>
      <?php include('encabezado.php'); ?>
      <h3>ELIMINACION DE DATOS DEL PRODUCTO</h3>
   </header>
   <section>
      <?php
        error_reporting(0);
        include 'conecta.php';
        include 'captura.php';
        $codigo = $_POST['txtCodigo'];
        $mensaje='';

        if (isset($_POST['btnMantenimiento'])){
            $boton=$_POST['btnMantenimiento'];

            if ($boton=="BUSCAR"){
               $rs=  mysql_query("CALL sp_buscaProducto('$codigo')",$cn);
               $n=  mysql_num_rows($rs);
               if ($n==0)
                   echo "<script>alert('Producto NO existe')</script>";

               $miProducto = mysql_fetch_array($rs);
             }

            if($boton=="ELIMINAR"){
              $cod= $codigo;
              $rs=  mysql_query("Call sp_eliminaProducto('$cod')",$cn);

              if ($rs==1)
                 echo "<script>
                           alert('Producto eliminado correctamente')
                           </script>";
              else
```

```
            echo "<script>
                      alert('Error al eliminar el producto')
                  </script>";
        }
      }
      ?>
      <form method="POST">
        <table border="1" width="550"
               cellspacing="1" cellpadding="1">
          <tr>
            <td>Ingrese codigo de producto</td>
            <td><input type="text" name="txtCodigo"
                       value="<?php echo $miProducto[0]; ?>" /></td>
            <td><input type="submit" name="btnMantenimiento"
                       value="BUSCAR" /><?php echo $mensaje; ?></td>
          </tr>
        </table>

        <table border="1" width="550"
               cellspacing="1" cellpadding="1">
         <tr>
           <td>DESCRIPCION</td>
           <td colspan="3">
             <input type="text" name="txtDescripcion"
                    value="<?php echo $miProducto[1]; ?>" />
           </td>
         </tr>
         <tr>
           <td>PRECIO</td>
           <td><input type="text" name="txtPrecio"
                      value="<?php echo $miProducto[2]; ?>" /></td>
           <td>STOCK</td>
           <td><input type="text" name="txtStock"
                      value="<?php echo $miProducto[3]; ?>" /></td>
         </tr>
        <tr>
           <td>FECHA VENC.</td>
           <td><input type="text" name="txtFecha"
                      value="<?php echo $miProducto[4]; ?>" /></td>
           <td colspan="2">
                 <input type="submit" name="btnMantenimiento"
                        value="ELIMINAR" /></td>
           </tr>
         </table>
    </form>
    </section>
    <footer></footer>
  </body>
  </html>
```

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
table{
    margin: auto;
}
```

```
img{
    margin: auto;
    display: block;
}
h3,h5{
    text-align: center;
}
#centrado{
    text-align: center;
}
td {
    border: solid 1px #006699;
    border-top: #006699;
    border-right: #006699;
    border-left: #006699;
    border-bottom: #006699;
}
```

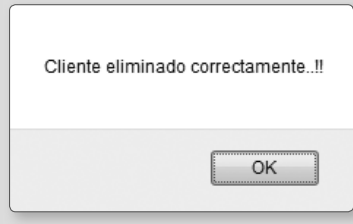
Caso desarrollado **Eliminando registro de cliente**

Implementar una aplicación web con PHP y MySQL que permita eliminar un registro de cliente desde la base de datos Ventas2015, tal como se muestra en la siguiente imagen:

Imagen: elimina.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **EliminaCliente**.
- Antes de eliminar los datos de un determinado cliente, se debe buscarlo por su código; una vez encontrado se deberán mostrar sus datos, de forma que el usuario pueda visualizar los datos del cliente a eliminar.
- Una vez mostrados los datos del cliente a eliminar, podrá realizar la eliminación definitiva seleccionando el botón **Eliminar**.
- Si la eliminación es realizada de manera correcta, deberá mostrar el mensaje «Cliente eliminado correctamente..!!» usando la función ALERT de Javascript.



- El pie de página debe mostrar los enlaces para listar y registrar los datos de los clientes, el listado mostrará todos los registros de los clientes, incluyendo sus imágenes:

Imagen: listado.php

El objetivo de cada archivo del proyecto es:

- **conecta.php**: Contiene las sentencias que permite conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **captura.php**: Contiene las funciones de captura de todos los valores registrados en el formulario.
- **elimina.php**: Contiene la página que permite buscar y eliminar los datos de un determinado cliente.
- **registra.php**: Contiene la página que permite registrar los datos del nuevo cliente.
- **listado.php**: Contiene la página que permite listar a los clientes con sus respectivas imágenes y a la vez para comprobar la eliminación de los datos del cliente.
- **miEstilo.css**: Contiene los estilos que se aplicarán a los archivos **listado.php** y **actualiza.php**.

Procedimientos almacenados

```
DROP PROCEDURE IF EXISTS sp_nuevocliente;
CREATE PROCEDURE sp_nuevocliente(    COD CHAR(5),NOM VARCHAR(30),
                                        PAT VARCHAR(30),MAT VARCHAR(30),
                                        DIR VARCHAR(40),TEL CHAR(15),
                                        DIS CHAR(3),COR VARCHAR(50))
    INSERT INTO cliente VALUES(cod,nom,pat,mat,dir,tel,dis,cor);

DROP PROCEDURE IF EXISTS sp_listadoclientes;
CREATE PROCEDURE sp_listadoclientes()
    SELECT C.ID_CLIENTE,
            CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
            C.DIRECCION,C.FONO,D.DESCRIPCION,C.EMAIL
            FROM CLIENTE C
            INNER JOIN DISTRITO D
            ON C.ID_DISTRITO=D.ID_DISTRITO
            ORDER BY 1 ASC;

DROP PROCEDURE IF EXISTS sp_buscacliente;
CREATE PROCEDURE sp_buscacliente(cod char(5))
    SELECT * FROM CLIENTE
                    WHERE ID_CLIENTE=COD;

DROP PROCEDURE IF EXISTS sp_eliminacliente;
CREATE PROCEDURE sp_eliminacliente(cod char(5))
    DELETE FROM CLIENTE
                    WHERE ID_CLIENTE=COD;
```

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **captura.php**

```
<?php
    function getCodigo(){
        return $_POST['txtCodigo'];
    }
?>
```

Archivo: **elimina.php**

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title></title>
  <link href="miEstilo.css" rel="stylesheet">
</head>
<body>
  <header>
    <h3>ELIMINACION DE DATOS DEL CLIENTE</h3>
    <img src="banner_clientes.jpg" width="550" height="180"/>
  </header>
  <section>
    <?php
      error_reporting(0);
      include('conecta.php');
      include('captura.php');

      $codigo=  getCodigo();

      if (isset($_POST['btnGenerico'])){
         $boton=$_POST['btnGenerico'];

         if ($boton == 'BUSCAR'){
           $rs=mysql_query("CALL sp_buscacliente('$codigo');",$cn);
           $cliente = mysql_fetch_array($rs);
         }

         if ($boton=='ELIMINAR'){
           $rs=  mysql_query("CALL sp_eliminacliente('$codigo')",$cn);
           if ($rs)
              echo "<script>
                      alert ('Cliente eliminado correctamente..!!');
                      history.back();
                    </script>";
            else
              echo '<script>
                 alert ("Error al eliminar al cliente - '.mysql_error().'");
                  history.back();
                    </script>';
          }
      }
    ?>

    <form method="POST">
      <table border="0" width="550" cellspacing="1" cellpadding="10">
        <tr>
          <td>Ingrese codigo</td>
          <td><input type="text" name="txtCodigo" maxlength="5"
                     value="<?php echo getCodigo(); ?>" /></td>
          <td><input type="submit" name="btnGenerico" value="BUSCAR" /></td>
        </tr>
      </table>
      <br>
      <table border="0" width="550" cellspacing="1" cellpadding="1">
        <tr>
          <th>NOMBRES</th>
          <td colspan="3"><?php echo $cliente[1] ?></td>
```

```
        </tr>
        <tr>
          <th>PATERNO</th>
          <td><?php echo $cliente[2] ?></td>
          <th>MATERNO</th>
          <td><?php echo $cliente[3] ?></td>
        </tr>
        <tr>
          <th>DIRECCION</th>
          <td colspan="3"><?php echo $cliente[4] ?></td>
        </tr>
        <tr>
          <th>TELEFONO</th>
          <td><?php echo $cliente[5] ?></td>
          <th>DISTRITO</th>
          <td><?php echo $cliente[6];?></td>
        </tr>
        <tr>
          <th>EMAIL</th>
          <td colspan="3"><?php echo $cliente[7] ?></td>
        </tr>
        <tr>
          <td colspan="4">
             <input type="submit" name="btnGenerico" value="ELIMINAR" />
          </td>
        </tr>
      </table>
    </form>
  </section>
  <footer>
        <h5>Todos los derechos reservados @2015</h5>
        <h5>
            <a href="listado.php">LISTADO DE CLIENTES </a> |
            <a href="elimina.php">ELIMINA DATOS DE CLIENTES </a> |
            <a href="registra.php">REGISTRO DEL NUEVO CLIENTE </a>
        </h5>
  </footer>
</body>
</html>
```

Comentarios

```
<?php
  error_reporting(0);
  include('conecta.php');
  include('captura.php');

  $codigo=  getCodigo();

  if (isset($_POST['btnGenerico'])){
     $boton=$_POST['btnGenerico'];

     if ($boton == 'BUSCAR'){
       $rs=mysql_query("CALL sp_buscacliente('$codigo');",$cn);
       $cliente = mysql_fetch_array($rs);
     }

     if ($boton=='ELIMINAR'){
       $rs=  mysql_query("CALL sp_eliminacliente('$codigo')",$cn);
       if ($rs)
         echo "<script>
```

```
                alert ('Cliente eliminado correctamente..!!');
                history.back();
              </script>";
          else
           echo '<script>
           alert ("Error al eliminar al cliente - '.mysql_error().'");
           history.back();
           </script>';
       }
  }
 ?>
```

Se inicia con la function **error_reporting** para bloquear los errores de advertencia de PHP, seguidamente se agregan las variables de conexión al servidor y selección de la base de datos Ventas2015 desde el archivo **conecta.php**. Además de incluir las funciones que permiten capturar los valores ingresados en los controles del formulario, mediante el archivo **captura.php**.

A partir de aquí, se obtiene el código del cliente, tanto para la búsqueda como para la eliminación del cliente. Como ya se ha visto en casos anteriores, los botones implementados tienen el mismo nombre para poder identificarlos por su contenido y por problemas de submit dentro del mismo formulario, es así que se pregunta si el botón btnGenerico ha sido seleccionado, y se captura qué botón ha sido seleccionado en la variable **$boton**, luego se evalúa el tipo de botón; en caso sea BUSCAR, se invoca al procedimiento almacenado de búsqueda SP_BUSCACLIENTE, el cual envía el resultado de la consulta a la variable **$rs**, como la búsqueda emite solo un registro como respuesta, este puede ser capturado mediante una variable tipo arreglo llamada **$cliente**, como observa, aquí no se debe usar la estructura repetitiva ya que se trata de un solo registro.

En caso seleccione el botón **ELIMINAR**, se invoca al procedimiento almacenado SP_ELIMINACLIENTE, cuya respuesta positiva o no, es enviada a la variable **$rs**; esto se da con la intención de poder mostrar los mensajes solicitados por el caso, cuando el cliente sea eliminado correctamente o no, tenga en cuenta que la función alter solo funciona dentro de las etiquetas <SCRIPT> y que la sentencia **history.back** devuelve a la página anterior.

```
<script>
          alert ("Error al eliminar al cliente - '.mysql_error().'");
          history.back();
</script>
```

```
<tr>
    <th>NOMBRES</th>
    <td colspan="3"><?php echo $cliente[1] ?></td>
</tr>
```

Se imprimirán dentro de los controles correspondientes los valores encontrados en el proceso de búsqueda, tal es así que **$cliente** tiene todos los valores; **$cliente[1]** representa al nombre del cliente.

Archivo: **registra.php**

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title></title>
```

```
    <link href="miEstilo.css" rel="stylesheet">
</head>
<body>
  <header>
    <h3>REGISTRO DEL NUEVO CLIENTE</h3>
    <img src="banner_clientes.jpg" width="700" height="250"/>
  </header>
  <section>
    <?php
      include('conecta.php');
      if (isset($_POST['btnRegistrar'])){
       $codigo=$_POST['txtCodigo'];
       $nombres=$_POST['txtNombres'];
       $paterno=$_POST['txtPaterno'];
       $materno=$_POST['txtMaterno'];
       $direccion=$_POST['txtDireccion'];
       $telefono=$_POST['txtFono'];
       $distrito=$_POST['selDistrito'];
       $correo=$_POST['txtCorreo'];

       $rs=  mysql_query("CALL sp_nuevocliente('$codigo','$nombres',
                                               '$paterno','$materno',
                                               '$direccion','$telefono',
                                               '$distrito','$correo')",$cn);
       if ($rs)
         echo "<script>
                 alert ('Cliente registrado correctamente..!!');
                 history.back();
               </script>";
        else
         echo '<script>
                 alert ("Error al registrar al cliente - '.mysql_error().'");
                 history.back();
               </script>';
      }
    ?>
    <form method="POST">
      <table border="0" width="700" cellspacing="0" cellpadding="0">
         <tr>
            <td width="500"></td>
            <td>CODIGO
               <input type="text" name="txtCodigo"
                      maxlength="5" size="45"/>
             </td>
         </tr>
         <tr>
            <td colspan="2">NOMBRES</td>
         </tr>
         <tr>
            <td colspan="2">
                  <input type="text" name="txtNombres" size="112"/>
            </td>
         </tr>
         <tr>
            <td>PATERNO</td>
            <td>MATERNO</td>
         </tr>
         <tr>
            <td><input type="text" name="txtPaterno" size="45"/></td>
            <td><input type="text" name="txtMaterno" size="45"/></td>
         </tr>
         <tr>
            <td colspan="2">DIRECCION</td>
         </tr>
```

```
          <tr>
             <td colspan="2">
                 <input type="text" name="txtDireccion" size="112"/></td>
          </tr>
          <tr>
             <td>FONO</td>
             <td>DISTRITO</td>
          </tr>
          <tr>
             <td><input type="text" name="txtFono" size="45"/></td>
             <td><select name="selDistrito">
                    <?php
                       $rs=mysql_query('CALL SP_DISTRITOS',$cn);
                       $n=mysql_num_rows($rs);

                       while($distritos = mysql_fetch_array($rs)){
                           echo "<option value='".$distritos[0]."'>"
                                .$distritos[1]."</option>";
                       }
                    ?>
                 </select>
             </td>
           </tr>
           <tr>
             <td>CORREO ELECTRONICO</td>
             <td></td>
           </tr>
           <tr>
             <td colspan="2"><input type="text" name="txtCorreo"
                                    size="112"/></td>
           </tr>
           <tr>
             <td><input type="submit" value="REGISTRAR" name="btnRegistrar" />
             </td>
             <td></td>
           </tr>
        </table>
    </form>
  </section>
  <footer>
     <h5>Todos los derechos reservados @2015</h5>
     <h5>
        <a href="listado.php">LISTADO DE CLIENTES </a> |
        <a href="elimina.php">ELIMINA DATOS DE CLIENTES </a> |
        <a href="registra.php">REGISTRO DEL NUEVO CLIENTE </a>
      </h5>
  </footer>
</body>
</html>
```

Comentarios

```
<?php
   include('conecta.php');
   if (isset($_POST['btnRegistrar'])){
     $codigo=$_POST['txtCodigo'];
     $nombres=$_POST['txtNombres'];
     $paterno=$_POST['txtPaterno'];
     $materno=$_POST['txtMaterno'];
     $direccion=$_POST['txtDireccion'];
     $telefono=$_POST['txtFono'];
     $distrito=$_POST['selDistrito'];
```

```
      $correo=$_POST['txtCorreo'];

      $rs=  mysql_query("CALL sp_nuevocliente('$codigo','$nombres',
                                              '$paterno','$materno',
                                              '$direccion','$telefono',
                                              '$distrito','$correo')",$cn);
      if ($rs)
        echo "<script>
        alert ('Cliente registrado correctamente..!!');
        history.back();
        </script>";
        else
        echo '<script>
          alert ("Error al registrar al cliente - '.mysql_error().'");
          history.back();
        </script>';
    }
?>
```

Se incluyen las variables de conexión al servidor y la base de datos, seguidamente se identifica si el botón btnRegistrar se ha presionado, para capturar los valores desde el formulario ya que estos serán enviados al procedimiento almacenado SP_NUEVOCLIENTE; el cual permite registrar los datos del nuevo cliente.

```
<td><select name="selDistrito">
       <?php
         $rs=mysql_query('CALL SP_DISTRITOS',$cn);
         $n=mysql_num_rows($rs);

         while($distritos = mysql_fetch_array($rs)){
           echo "<option value='".$distritos[0]."'>"
                                  .$distritos[1]."</option>";
         }
       ?>
   </select>
</td>
```

El script permite enviar los distritos al control cuadro combinado desde el procedimiento almacenado SP_DISTRITOS, tenga en cuenta que **$distritos[0]** representa al código y **$distritos[1]** al nombre del distrito.

Archivo: **listado.php**

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">

</head>
<body>
    <?php
        include('conecta.php');
    ?>
  <header>
    <h3>LISTADO DE CLIENTES</h3>
```

```
    <img src="banner_clientes.jpg" width="800" height="250">
  </header>
  <section>
    <table width="800" border="0" cellpadding="0" cellspacing="0">
    <tr>
            <th width="80">CODIGO</th>
            <th width="280">CLIENTE</th>
            <th width="280">DIRECCION</th>
            <th width="160">TELEFONO</th>
            <th width="160">DISTRITO</th>
            <th width="100">IMAGEN</th>
    </tr>
      <?php
            $rs=mysql_query("CALL SP_LISTADOCLIENTES",$cn);
            while($clientes = mysql_fetch_array($rs)){
    ?>
    <tr>
            <td><?php echo $clientes[0];?></td>
            <td><?php echo $clientes[1];?></td>
            <td><?php echo $clientes[2];?></td>
            <td><?php echo $clientes[3];?></td>
            <td><?php echo $clientes[4];?></td>
            <td>
            <?php
                //Declarando el nombre del archivo a buscar
                $archivo = "fotos/".$clientes[0].".jpg";

                //Buscando el archivo
                if (file_exists($archivo))
                    echo "<img src=fotos/".$clientes[0].".jpg width=60>";
                else
                    echo "<img src=fotos/sinfoto.jpg width=60>";
            ?>
            </td>
    </tr>
        <?php
            }
      ?>
    </table>
</section>
<footer>
        <h5>Todos los derechos reservados @2015</h5>
        <h5>
            <a href="listado.php">LISTADO DE CLIENTES </a> |
            <a href="elimina.php">ELIMINA DATOS DE CLIENTES </a> |
            <a href="registra.php">REGISTRO DEL NUEVO CLIENTE </a>
        </h5>
</footer>
</body>
</html>
```

```
<?php
   include('conexion.php');
?>
```

Se incluyen dentro del archivo las variables de conexión al servidor y la selección de la base de datos Ventas2015.

```
<?php
   $rs=mysql_query("CALL SP_LISTADOCLIENTES",$cn);
   while($clientes = mysql_fetch_array($rs)){
?>
```

Se invoca al procedimiento almacenado SP_LISTADOCLIENTES que permite mostrar los datos de los clientes para el listado, esto es enviado a la variable **$rs**. La estructura **while** recorre por todos los clientes para posteriormente imprimirlos.

```
<tr>
   <td><?php echo $clientes[0];?></td>
   <td><?php echo $clientes[1];?></td>
   <td><?php echo $clientes[2];?></td>
   <td><?php echo $clientes[3];?></td>
   <td><?php echo $clientes[4];?></td>
   <td>
      <?php
         $archivo = "fotos/".$clientes[0].".jpg";

         if (file_exists($archivo))
            echo "<img src=fotos/".$clientes[0].".jpg width=60>";
            else
            echo "<img src=fotos/sinfoto.jpg width=60>";
      ?>
    </td>
</tr>
```

Finalmente, se imprimirá el resultado obtenido en la variable **$clientes** en cada una de las celdas de la tabla. Además, la imagen de cliente debe ser mostrada mediante el código del cliente obtenido desde la carpeta **Fotos** del proyecto, en caso de no tener foto se mostrará la imagen **sinfoto.jpg**.

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
table {
    margin: auto;
    background-color: antiquewhite;
}
img{
    margin: auto;
    display: block;
}
table th {
    background-color: #b9c9fe;
    text-align: left;
```

3.4 Casos desarrollados

Caso desarrollado 1 **Mantenimiento de registro de clientes**

Implementar una aplicación web con PHP y MySQL que permita realizar la gestión de mantenimiento de registros de clientes usando la base de datos Ventas2015, tal como se muestra en la siguiente imagen:

Imagen: principal.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **MantenimientoCliente**.
- Al iniciar la aplicación se debe mostrar el archivo **principal.php**, aquí se debe considerar que los nombres de los distritos se encuentran en el cuadro combinado precargados, así como los tipos de búsqueda que puede realizar el usuario, por ejemplo, «Por nombre del cliente» o por «Código de cliente».
- Para gestionar el mantenimiento de los datos del cliente, se debe contar con botones para dicha actividad, el cual se detalla a continuación:

  - **LISTAR:** Permite listar los clientes que ya se encuentran registrados en la base de datos, esta información será mostrada en el pie de página del archivo **principal.php**, tal como se muestra en la siguiente imagen:

CODIGO	CLIENTE	DIRECCION	TELEFONO	CORREO ELECTRONICO
C0001	CARLOS LOPEZ HURTADO	Calle El Pino 346	4677352	CERCADO
C0002	FRIDA QUIROZ DE LA CRUZ	Jr. Los Reyes 475	6734732	CARABAYLLO
C0003	ALEJANDRO TRAUCO MANRIQUE	Jr.Huascar 956	4676732	BARRANCO
C0004	CARLA BLANCO RUIZ	Av. Arequipa 451	4867438	ATE
C0005	CORINA MENDOZA PEREZ	Av. Ingenieros 222	2548965	BREÑA
C0006	JORGE RODAS DIONICIO	Jr. Las Liras 456	4789658	ATE
C0007	ORLANDO CUEVAS CABANILLAS	Calle La Encantada 425	5698532	JESUS MARIA
C0008	ANTUANE RODRIGUEZ ALARCON	Av. Dorado 347	4589732	BREÑA
C0009	MANUEL SUAREZ FERNANDEZ	Jr. Los Robles 854	4576738	PACHACAMAC
C0010	JUAN GUTIERREZ DIAZ	Calle Girasoles 456	4897421	LINCE
C0011	CARLOS COLAN BARDALES	Av. Los Heroes 895	3698574	PACHACAMAC
C0012	MARTIN CARRILLO SALAS	Calle Los Huertos 844	8965952	CHACLACAYO
C0013	JOSE LAZARTE LUJAN	Jr. Agapito 452	1258965	MAGDALENA
C0014	VIDAL ZORRILLA RODRIGUEZ	Av. Heroes del cenepa 635	1547894	PUCUSANA
C0015	GUILLERMO RAMOS FLORES	Jr. Las Almendras 211	4587964	ATE
C0016	Gabriela Torres Lazaro	Av. Las Palmeras	5332523	JESUS MARIA

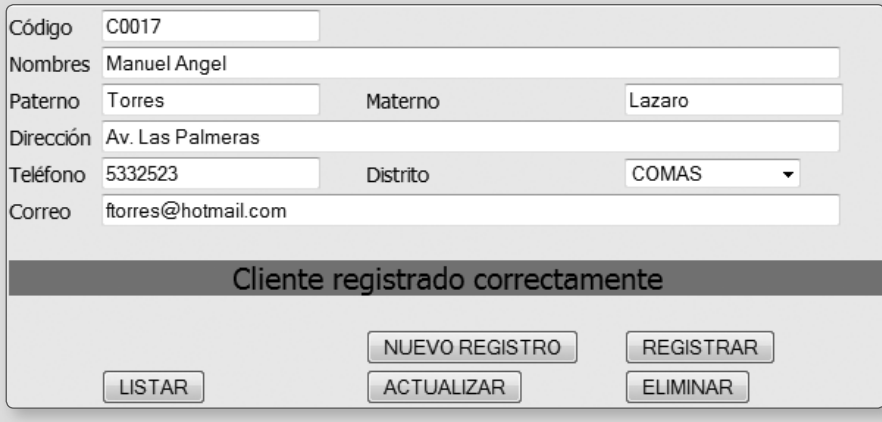
Imagen: principal.php

- **NUEVO REGISTRO:** Este botón tiene por misión autogenerar el nuevo código del cliente, tal como se muestra en la siguiente imagen:

Imagen: principal.php

Dicho código deberá ser mostrado únicamente al presionar el botón **NUEVO REGISTRO**, a la vez que debe mostrarse bloqueado, luego el usuario podrá registrar los datos del nuevo cliente, una vez llenado los controles podrá registrar los datos mediante el botón **REGISTRAR**.

Finalmente, se debe mostrar el mensaje «Cliente registrado correctamente», tal como se muestra en la siguiente imagen:



- **ACTUALIZAR:** Tiene por misión registrar los cambios realizados a un determinado cliente, para lo cual primero debe mostrarse los datos del mismo en los controles; esto se logrará cuando el usuario seleccione un tipo de búsqueda, ingrese un valor en el control y presione el botón **Buscar**.

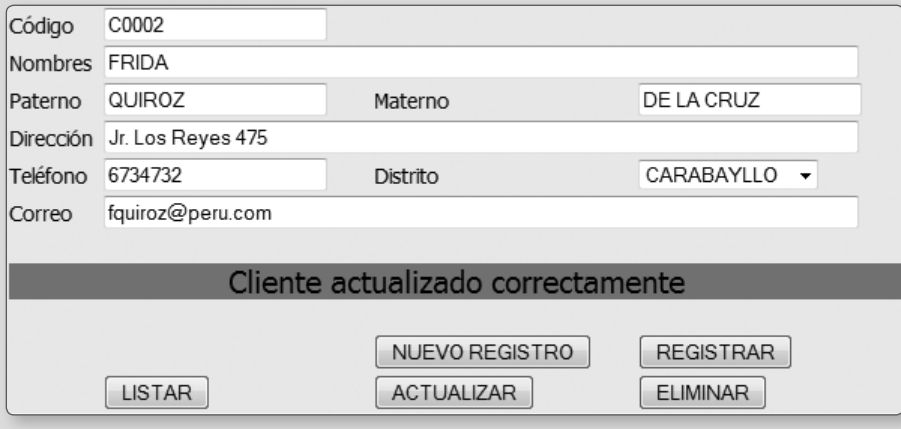


Imagen: principal.php

- **ELIMINAR:** Tiene por misión eliminar el registro de un determinado cliente, para lo cual primero debe mostrarse los datos del mismo en los controles; esto se logrará cuando el usuario seleccione un tipo de búsqueda, ingrese un valor en el control y presione el botón **Buscar**. Si todo es correcto, se debe mostrar el siguiente mensaje:

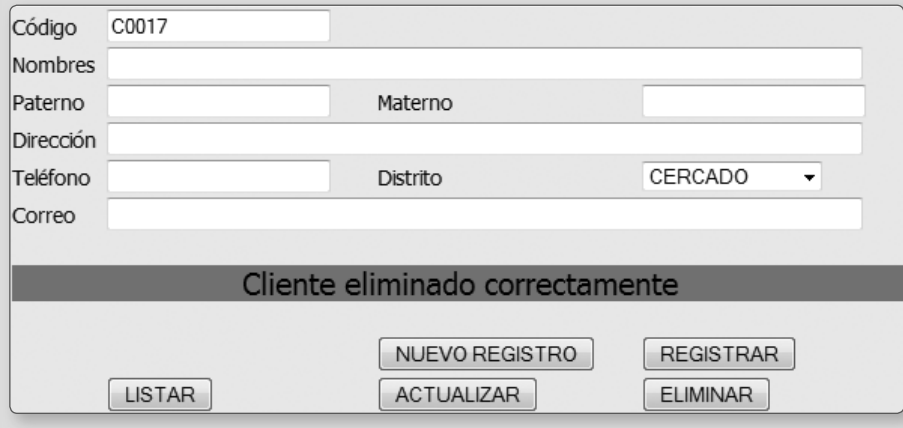


Imagen: principal.php

El objetivo de cada archivo del proyecto es:

- **conecta.php:** Contiene las sentencias que permite conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **captura.php:** Contiene las funciones de captura de todos los valores registrados en el formulario.
- **principal.php:** Contiene la página que interactúa con el usuario.
- **encabezado.php:** Contiene la imagen que se muestra como encabezado de página en todo el proyecto.
- **inicializador.php:** Permite inicializar todas las variables usadas en el proyecto con valores vacíos.
- **miEstilo.css:** Contiene los estilos que se aplicarán en el proyecto de actualización.

Procedimientos almacenados

```
DROP PROCEDURE IF EXISTS sp_nuevocliente;
CREATE PROCEDURE sp_nuevocliente(    COD CHAR(5),NOM VARCHAR(30),
                                     PAT VARCHAR(30),MAT VARCHAR(30),
                                     DIR VARCHAR(40),TEL CHAR(15),
                                     DIS CHAR(3),COR VARCHAR(50))
    INSERT INTO cliente VALUES(cod,nom,pat,mat,dir,tel,dis,cor);

DROP PROCEDURE IF EXISTS sp_listadoclientes;
CREATE PROCEDURE sp_listadoclientes()
    SELECT C.ID_CLIENTE,
           CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
           C.DIRECCION,C.FONO,D.DESCRIPCION,C.EMAIL
           FROM CLIENTE C
           INNER JOIN DISTRITO D
           ON C.ID_DISTRITO=D.ID_DISTRITO
           ORDER BY 1 ASC;


DROP PROCEDURE IF EXISTS sp_buscacliente;
CREATE PROCEDURE sp_buscacliente(cod char(5))
    SELECT * FROM CLIENTE
                  WHERE ID_CLIENTE=COD;
```

```
DROP PROCEDURE IF EXISTS sp_actualizacliente;
CREATE PROCEDURE sp_actualizacliente(COD CHAR(5),NOM VARCHAR(30),
                                     PAT VARCHAR(30),MAT VARCHAR(30),
                                     DIR VARCHAR(40),TEL CHAR(15),
                                     DIS CHAR(3),COR VARCHAR(50))
    UPDATE CLIENTE
            SET NOMBRES=NOM, PATERNO=PAT, MATERNO=MAT, DIRECCION=DIR,
                    FONO=TEL, ID_DISTRITO=DIS, EMAIL=COR
            WHERE ID_CLIENTE=COD;

DROP PROCEDURE IF EXISTS sp_eliminacliente;
CREATE PROCEDURE sp_eliminacliente(cod char(5))
    DELETE FROM CLIENTE
                    WHERE ID_CLIENTE=COD;
```

Archivo: **conecta.php**

```
<?php
    $cn=mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn)
?>
```

Archivo: **principal.php**

```
<!DOCTYPE html>
<html>
   <head>
     <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
     <title></title>
     <link href="miEstilo.css" rel="stylesheet">
   </head>
   <body>
     <header>
       <?php include('encabezado.php'); ?>
     </header>
     <?php
        //Activamos la conexion al servidor y
        //seleccionamos la base de datos
        include('conecta.php');

        //Invocamos a todas las funciones de captura de valor
        include('captura.php');

        //Inicializamos las variables para no ocasionar errores
        include('inicializador.php');

        //Si el usuario selecciona uno de los botones
        if (isset($_POST['btnMantenimiento'])){

        //Capturamos que tipo de boton selecciono
        $boton = $_POST['btnMantenimiento'];

        //Si el boton seleccionado es BUSCAR
        if ($boton=='BUSCAR'){
          //Capturamos el tipo de consulta y el valor registrado
          $tipo=  getTipo();
          $valor= getValor();

          //Si el tipo de consulta es por "Nombre de cliente"
          if ($tipo==0){
```

```
        $rs=  mysql_query("SELECT *
                FROM CLIENTE
                WHERE CONCAT(NOMBRES,' ',PATERNO,' ',MATERNO)
                                                LIKE '$valor'",$cn);
        $cliente=mysql_fetch_array($rs);
        $codigo=$cliente[0];
        $nombres=$cliente[1];
        $paterno=$cliente[2];
        $materno=$cliente[3];
        $direccion=$cliente[4];
        $telefono=$cliente[5];
        $distrito=$cliente[6];
        $correo=$cliente[7];
     }

     //Si el tipo de consulta es por "Codigo"
     if ($tipo==1){
        $rs=  mysql_query("SELECT *
                                FROM CLIENTE
                                WHERE ID_CLIENTE='$valor'",$cn);
        $cliente=  mysql_fetch_array($rs);
        $codigo=$cliente[0];
        $nombres=$cliente[1];
        $paterno=$cliente[2];
        $materno=$cliente[3];
        $direccion=$cliente[4];
        $telefono=$cliente[5];
        $distrito=$cliente[6];
        $correo=$cliente[7];
     }
  }

  //Si el boton seleccionado es NUEVO REGISTRO
  if ($boton=='NUEVO REGISTRO'){
      //Consulta que devuelve el ultimo codigo de cliente registrado
      $rs=  mysql_query('SELECT id_cliente
                              FROM cliente
                              ORDER BY id_cliente DESC LIMIT 1',$cn);

      //Capturando el ultimo codigo
      $ultimo=mysql_fetch_array($rs);

      //Generando el nuevo codigo
      $codigo='C'.substr((substr($ultimo[0],1)+10001),1);
    }

    //Si el boton seleccionado es REGISTRAR
    if ($boton=='REGISTRAR'){
       $codigo=  getCodigo();
       $nombres= getNombres();
       $paterno= getPaterno();
       $materno= getMaterno();
       $direccion= getDireccion();
       $telefono= getTelefono();
       $distrito = getDistrito();
       $correo= getCorreo();

       $rs=mysql_query("CALL SP_NUEVOCLIENTE('$codigo','$nombres',
                                          '$paterno','$materno',
                                          '$direccion','$telefono',
                                          '$distrito','$correo')",$cn);
       if ($rs==1)
         $mensaje='Cliente registrado correctamente';
         else {
```

```
            $mensaje = 'Error al registrar el cliente'.  mysql_error();
          }
        }

        //Si el boton seleccionado es ACTUALIZAR
        if ($boton=='ACTUALIZAR'){
           $codigo=  getCodigo();
           $nombres= getNombres();
           $paterno= getPaterno();
           $materno= getMaterno();
           $direccion= getDireccion();
           $telefono= getTelefono();
           $distrito = getDistrito();
           $correo= getCorreo();

          //Implementación de la sentencia de actualizacion del cliente
           $sql="CALL sp_actualizacliente('$codigo','$nombres',
                                          '$paterno','$materno',
                                          '$direccion','$telefono',
                                          '$distrito','$correo')";

            //Ejecución de la sentencia de actualizacion
            $rs=mysql_query($sql,$cn);
            if ($rs==1)
              $mensaje='Cliente actualizado correctamente';
              else {
            $mensaje='Ocurrio un problema al actualizar '.mysql_error();
          }
        }

        //Si el boton seleccionado es ELIMINAR
        if ($boton=='ELIMINAR'){
           $codigo=getCodigo();

           //Implementación de la sentencia de eliminacion del cliente
           $sql="CALL sp_eliminacliente('$codigo')";

           //Ejecución de la sentencia de eliminacion
           $rs=mysql_query($sql,$cn);
           $mensaje='Ocurrio un problema al eliminar '.mysql_error();
           if ($rs==1)
              $mensaje='Cliente eliminado correctamente';
              else {
              $mensaje = 'Error al eliminar el cliente'.  mysql_error();
        }
       }
    }
?>
   <section>
      <form method="POST">
        <table border="0" width="600" cellspacing="1" cellpadding="1">
           <tr>
             <td>SELECCIONE TIPO DE BUSQUEDA</td>
             <td><select name="selTipo">
                     <option value="0">Por nombre de cliente</option>
                      <option value="1">Por codigo</option>
                 </select></td>
             <td></td>
           </tr>
           <tr>
             <td>INGRESE VALOR</td>
            <td><input type="text" name="txtCriterio" value="" /></td>
             <td><input type="submit" name="btnMantenimiento"
                        value="BUSCAR" /></td>
```

```
    </tr>
</table>
<br>
<table border="0" width="600" cellspacing="1" cellpadding="1">
    <tr>
        <td>Codigo</td>
        <td><input type="text" name="txtCodigo"
                   value="<?php echo $codigo; ?>" /></td>
        <td></td>
        <td></td>
    </tr>
    <tr>
        <td>Nombres</td>
        <td colspan="3"><input type="text" name="txtNombres"
                          value="<?php echo $nombres; ?>" size="79"/>
        </td>
    </tr>
    <tr>
      <td>Paterno</td>
      <td><input type="text" name="txtPaterno"
                 value="<?php echo $paterno; ?>" /></td>
      <td>Materno</td>
      <td><input type="text" name="txtMaterno"
                 value="<?php echo $materno; ?>" /></td>
    </tr>
    <tr>
        <td>Direccion</td>
        <td colspan="3"><input type="text" name="txtDireccion"
                value="<?php echo $direccion; ?>" size="79"/>
        </td>
    </tr>
    <tr>
        <td>Telefono</td>
        <td><input type="text" name="txtTelefono"
                   value="<?php echo $telefono; ?>" /></td>
        <td>Distrito</td>
        <td><select name="selDistrito">
             <?php
                //Generando la consulta de Distritos
                $rsD=  mysql_query('SELECT * FROM DISTRITO',$cn);

                //Recorrer por todos los distritos
                while($miDistrito = mysql_fetch_array($rsD)){

                   //Activar SELECTED segun el distrito del cliente
                   if ($miDistrito[0]==$distrito)
                      $seleccionado='SELECTED';
                      else
                      $seleccionado='';
             ?>
             <option value="<?php echo $miDistrito[0];?>"
                           <?php echo $seleccionado; ?>>
                           <?php echo $miDistrito[1]; ?>
             </option>
             <?php
               } //Fin del while
             ?>
            </select></td>
    </tr>
    <tr>
      <td>Correo</td>
      <td colspan="3"><input type="text" name="txtCorreo"
                          value="<?php echo $correo; ?>" size="79"/>
      </td>
```

```
            </tr>
            <tr>
            <td colspan="4"><p id="centrado"><?php echo $mensaje; ?></p></td>
            </tr>
            <tr>
              <td></td>
              <td></td>
              <td><input type="submit" name="btnMantenimiento"
                         value="NUEVO REGISTRO" /></td>
              <td><input type="submit" name="btnMantenimiento"
                         value="REGISTRAR" /></td>
            </tr>
            <tr>
              <td></td>
              <td><input type="submit" name="btnMantenimiento"
                         value="LISTAR" /></td>
              <td><input type="submit"
                         name="btnMantenimiento" value="ACTUALIZAR" /></td>
              <td><input type="submit" name="btnMantenimiento"
                         value="ELIMINAR" /></td>
            </tr>
          </table>
          <br>
          <?php
             if ($boton=='LISTAR'){
          ?>
             <table border="0" width="600" cellspacing="1" cellpadding="1">
               <tr>
                  <th>CODIGO</th>
                  <th>CLIENTE</th>
                  <th>DIRECCION</th>
                  <th>TELEFONO</th>
                  <th>CORREO ELECTRONICO</th>
               </tr>
               <?php
                  //Generando la consulta para el listado de clientes
                  $rs=  mysql_query('call sp_listadoclientes()',$cn);

                  //Recorriendo todos los registros de clientes
                  while($misClientes = mysql_fetch_array($rs)){
               ?>
                   <tr>
                       <td id="listado"><?php echo $misClientes[0] ?></td>
                       <td id="listado"><?php echo $misClientes[1] ?></td>
                       <td id="listado"><?php echo $misClientes[2] ?></td>
                       <td id="listado"><?php echo $misClientes[3] ?></td>
                       <td id="listado"><?php echo $misClientes[4] ?></td>
                   </tr>
                   <?php
                         } //Fin del while
                       }
                   ?>
                   </table>
             </form>
         </section>
         <footer> </footer>
      </body>
  </html>
```

Archivo: **captura.php**

```
<?php
    function getTipo(){
        return $_POST['selTipo'];
    }
    function getValor(){
        return $_POST['txtCriterio'];
    }
    function getCodigo(){
        return $_POST['txtCodigo'];
    }
    function getNombres(){
        return $_POST['txtNombres'];
    }
    function getPaterno(){
        return $_POST['txtPaterno'];
    }
    function getMaterno(){
        return $_POST['txtMaterno'];
    }
    function getDireccion(){
        return $_POST['txtDireccion'];
    }
    function getTelefono(){
        return $_POST['txtTelefono'];
    }
    function getDistrito(){
        return $_POST['selDistrito'];
    }
    function getCorreo(){
        return $_POST['txtCorreo'];
    }
?>
```

Archivo: **encabezado.php**

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <img src="banner.jpg" width="600" height="120"/>
    </body>
</html>
```

Archivo: **inicializador.php**

```
<?php
    $codigo='';
    $nombres='';
    $paterno='';
    $materno='';
    $direccion='';
    $telefono='';
    $distrito='';
```

```
    $correo='';
    $boton='';
    $mensaje='';
?>
```

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
#centrado{
    text-align: center;
    font-size: 20px;
    background-color: #FF3300;
}
table {
    margin: auto;
    background-color: antiquewhite;
}
img{
    margin: auto;
    display: block;
}
table th {
    font-size: 10px;
    background-color: #b9c9fe;
}

#listado{
    font-family: tahoma;
    font-size: 10px;
}

#rojo{
    text-align: center;
    font-size: 20px;
}
```

Caso desarrollado 2 Mantenimiento de registros de clientes paginado

Implementar una aplicación web con PHP y MySQL que permita realizar un mantenimiento a los registros de los clientes en la base de datos Ventas2015, se debe considerar que los clientes se deben mostrar paginados, tal como se muestra en la siguiente imagen:

CÓDIGO	CLIENTE	DIRECCIÓN	TELÉFONO	DISTRITO	CORREO ELECTRÓNICO	IMAGEN		
C0001	CARLOS LOPEZ HURTADO	Av. El Sol	4855959	CHORRILLOS	cclopez@gmail.com		ACTUALIZA	ELIMINA
C0002	FRIDA SOFIA QUIROZ DUARTE	Jr. Los Reyes 475	55-555-555 6	LURIGANCHO	fquiroz@hotmail.com		ACTUALIZA	ELIMINA
C0003	ALEJANDRO TRAUCO MANRIQUE	Jr.Huascar 956	4676732	BARRANCO	atruco@gmail.com		ACTUALIZA	ELIMINA
C0004	CARLA JULIA BLANCO RUIZ	Av. Arequipa 451	4867438	ATE	cblanco@hotmail.com		ACTUALIZA	ELIMINA

Página Nº 1 de 4
[1] [2] [3] [4]

Todos los derechos reservados @2015-ManuelTorres.com

Página: listado.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **ActualizaClientePaginado**.
- Al iniciar la aplicación se debe mostrar una lista de los clientes paginado, mostrando así cuatro clientes por página.
- Los datos que debe mostrar en el listado son código, nombres completos, dirección, teléfono, distrito, correo electrónico, imagen del cliente y las opciones actualiza y elimina, respectivamente.
- Se debe contar con una carpeta llamada **Fotos** que contenga las imágenes de los clientes, tal como se muestra a continuación:

- Al seleccionar el enlace **ACTUALIZA**, se deberá mostrar el archivo **actualiza.php** con los datos del cliente seleccionado, tal como se muestra en la siguiente imagen:

Página: actualiza.php

Al iniciar la página **actualiza.php** se mostrarán los datos del cliente en sus respectivos controles para que puedan ser modificados, tenga en cuenta que el código del cliente debe estar protegido, es decir, no podrá ser modificado.

Una vez modificado los datos necesarios del cliente, se presionará el botón **ACTUALIZAR** para registrar los cambios en la tabla; si se decide no realizar modificación alguna, se puede presionar el enlace «Retornar al listado» para retornar a la página **listado.php**.

Al final deberá mostrar el mensaje «Cliente actualizado correctamente» cuando la actualización ha sido correcta, en caso contrario mostrará el mensaje «Error al actualizar al cliente» además del mensaje obtenido desde la función **mysql_error**.

- Al seleccionar el enlace ELIMINA se deberá mostrar el archivo **elimina.php** con los datos del cliente seleccionado, tal como se muestra en la siguiente imagen:

Página: elimina.php

Al iniciar la página **elimina.php** se mostrarán los datos del cliente en una tabla así como su imagen.

Para eliminar al cliente se deberá presionar el botón **ACTUALIZAR** y así se registrarán los cambios en la tabla, si se decide no realizar modificación alguna se puede presionar el enlace «Retornar al listado» para retornar a la página **listado.php**.

Al final deberá mostrar el mensaje «Cliente actualizado correctamente», cuando la actualización ha sido correcta; en caso contrario, mostrará el mensaje «Error al actualizar al cliente», además del mensaje obtenido desde la función **mysql_error**.

El objetivo de cada archivo del proyecto es:

- **conecta.php:** Contiene las sentencias que permite conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **encabezado.php:** Contiene el formato del encabezado de todas las páginas del proyecto.
- **pie.php:** Contiene el formato del pie de todas las páginas del proyecto.
- **listado.php:** Contiene la página que permite listar los clientes de forma paginada, además de mostrar los enlaces de actualización y eliminación.
- **actualiza.php:** Muestra los datos del cliente que se requiere realizar alguna modificación.
- **grabaActualizacion.php:** Permite registrar los cambios realizados a un determinado cliente.
- **elimina.php:** Muestra los datos del cliente a eliminar, se debe tener en cuenta que los datos mostrados son de lectura.
- **grabaEliminacion.php:** Permite registrar la eliminación en la tabla clientes mediante un procedimiento almacenado.
- **miEstilo.css:** Contiene los estilos que se aplicarán a los archivos **listado.php** y **actualiza.php**.

Procedimientos almacenados

```
DROP PROCEDURE IF EXISTS sp_nuevocliente;
CREATE PROCEDURE sp_nuevocliente(    COD CHAR(5),NOM VARCHAR(30),
                                          PAT VARCHAR(30),MAT VARCHAR(30),
                                          DIR VARCHAR(40),TEL CHAR(15),
                                          DIS CHAR(3),COR VARCHAR(50))
    INSERT INTO cliente VALUES(cod,nom,pat,mat,dir,tel,dis,cor);

DROP PROCEDURE IF EXISTS sp_listadoclientes;
CREATE PROCEDURE sp_listadoclientes()
    SELECT C.ID_CLIENTE,
            CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
            C.DIRECCION,C.FONO,D.DESCRIPCION,C.EMAIL
            FROM CLIENTE C
            INNER JOIN DISTRITO D
            ON C.ID_DISTRITO=D.ID_DISTRITO
            ORDER BY 1 ASC;


DROP PROCEDURE IF EXISTS sp_buscacliente;
CREATE PROCEDURE sp_buscacliente(cod char(5))
    SELECT * FROM CLIENTE
                    WHERE ID_CLIENTE=COD;

DROP PROCEDURE IF EXISTS sp_actualizacliente;
CREATE PROCEDURE sp_actualizacliente(COD CHAR(5),NOM VARCHAR(30),
                                          PAT VARCHAR(30),MAT VARCHAR(30),
                                          DIR VARCHAR(40),TEL CHAR(15),
                                          DIS CHAR(3),COR VARCHAR(50))
    UPDATE CLIENTE
            SET NOMBRES=NOM, PATERNO=PAT, MATERNO=MAT, DIRECCION=DIR,
                    FONO=TEL, ID_DISTRITO=DIS, EMAIL=COR
            WHERE ID_CLIENTE=COD;

DROP PROCEDURE IF EXISTS sp_eliminacliente;
CREATE PROCEDURE sp_eliminacliente(cod char(5))
    DELETE FROM CLIENTE
                    WHERE ID_CLIENTE=COD;
```

Archivo: **conecta.php**

```
<?php
    $cn=mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn)
?>
```

Archivo: **encabezado.php**

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
      <img src="banner_clientes.jpg" width="900" height="250"/>
    </body>
</html>
```

Archivo: **pie.php**

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <h5>Todos los derechos reservados @2015-ManuelTorres.com</h5>
    </body>
</html>
```

Archivo: **listado.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
    <link href="estilo.css" rel="stylesheet">
  </head>
  <body>
    <header>
        <?php include('encabezado.php'); ?>
    </header>
    <section>
      <table border="0" width="900" cellspacing="1" cellpadding="5">
       <tr>
         <th>CODIGO</th>
         <th width="150">CLIENTE</th>
         <th>DIRECCION</th>
         <th>TELEFONO</th>
         <th>DISTRITO</th>
         <th>CORREO ELECTRONICO</th>
         <th>IMAGEN</th>
         <th></th>
         <th></th>
       </tr>
       <?php
         include('conecta.php');

       //Determinar el total de registros a mostrar
         $filas=4;

         //Condicionamos la existencia de la variable numero de pagina
         //cuando la variable no existe se inicia en la pagina 0, caso
         //se actualize dicha variable con el numero de pagina actual
         if (!isset($_GET["nPag"])) $nPag=0; else $nPag=$_GET["nPag"];

         //La variable $inicio determina el punto de inicio de los registros
         $inicio=$nPag*$filas;

         //Consulta que permite listar por bloques de 4 a los clientes
         $rs=mysql_query("SELECT ID_CLIENTE,
                          CONCAT(NOMBRES,' ',PATERNO,' ',MATERNO),
                          C.DIRECCION,C.FONO,D.DESCRIPCION,C.EMAIL
                          FROM CLIENTE C
                          INNER JOIN DISTRITO D
                          ON C.ID_DISTRITO=D.ID_DISTRITO
```

```
                        LIMIT $inicio,$filas",$cn);

    //Determinar el total de clientes
    $n=mysql_num_rows($rs);

   //Recorrer por los registros segun la consulta realizada
    while($clientes = mysql_fetch_array($rs)){
?>
<tr>
  <td><?php echo $clientes[0]; ?></td>
  <td><?php echo $clientes[1]; ?></td>
  <td><?php echo $clientes[2]; ?></td>
  <td><?php echo $clientes[3]; ?></td>
  <td><?php echo $clientes[4]; ?></td>
  <td><?php echo $clientes[5]; ?></td>
  <td>
    <?php
      //Mostrando la imagen del cliente
      echo "<img src=fotos/".$clientes[0].".jpg width=60>";
    ?>
  </td>
  <td><img src="lapiz.gif"/>
        <a href="actualiza.php?id=<?php echo $clientes[0]; ?>">
            ACTUALIZA
         </a></td>
   <td><img src="borrar.gif"/>
         <a href="elimina.php?id=<?php echo $clientes[0]; ?>">
              ELIMINA
         </a></td>
 </tr>
 <?php } ?>
 <tr id="centrado">
   <td colspan="9">
     <?php
        //Implementar la consulta de clientes
        $rsC=mysql_query("SELECT * FROM CLIENTE",$cn);

        //Determinar el total de registros
        $num=mysql_num_rows($rsC);

        //Determinar el total de paginas
        //(Total de Registros/Registros a mostrar)
        $totalPag=floor($num/$filas);

        //Si en la ultima hoja solo hay un registro este genera una
        //nueva pagina aumentando asi el total de paginas.
        if(($num%$filas)>0) $totalPag++;

        //Imprimiendo el numero actual y el total de paginas
        echo "Pagina Nº ".($nPag+1)." de ".$totalPag;
     ?>

     <br>
     <?php
        //Generando el enlace a las demas hojas de la paginación
        for($i=0;$i<$totalPag;$i++){
           if ($i==$nPag){
              echo "[".($i+1)."]";
           } else {
     ?>
     <a href='<?php echo $_SERVER['PHP_SELF']."?nPag=$i"?>'>
```

```
                        <?php echo "[".($i+1)."]" ?> </a>
            <?php
                  } //Cierre de else
                } //Cierre del for
            ?>
                </td>
          </tr>
        </table>
    </section>
    <footer>
       <?php include('pie.php'); ?>
    </footer>
 </body>
</html>
```

Archivo: **actualiza.php**

```
<!DOCTYPE html>
<html>
 <head>
   <meta charset="UTF-8">
   <title></title>
   <link href="estilo.css" rel="stylesheet">
 </head>
 <body>
   <header>
     <?php include('encabezado.php'); ?>
     <h3>ACTUALIZACION DE DATOS DEL CLIENTE</h3>
   </header>
   <section>
     <?php
       //Incluimos las variables de conexion al servidor y a la base de datos
       include('conecta.php');

       //Capturamos el codigo id enviado desde la pagina de
       //listado de clientes, la mision es determinar a que cliente se le
       //actualizara sus datos
       $codigo=$_GET['id'];

       //Obtenemos los nombres de los distritos en la variable $rsD
       $rsD=mysql_query('SELECT * FROM DISTRITO',$cn);

       //Obtenemos los datos exactos de un determinado cliente
       $rs=mysql_query("CALL sp_busca_cliente('$codigo')",$cn);

       //Enviamos la informacion obtenida en el $rs a la variable $cliente
       //para poder imprimirlo en las celdas correspondientes
       $cliente=  mysql_fetch_array($rs);
     ?>
     <form method="POST" action="grabaActualizacion.php">
       <table border="1" width="900" cellspacing="1" cellpadding="1">
        <tr>
          <th>CODIGO</th>
          <td><input type="text" name="txtCodigo"
                     value="<?php echo $cliente[0]; ?>" readonly /></td>
          <td rowspan="6">
             <?php
                //Mostrando la imagen del cliente
```

```
            echo "<img src=fotos/".$cliente[0].".jpg width=120>";
          ?>
        </td>
      </tr>
      <tr>
        <th>NOMBRES</th>
        <td><input type="text" name="txtNombres" size="80"
                   value="<?php echo $cliente[1]; ?>" /></td>
        <td></td>
      </tr>
      <tr>
        <th>PATERNO</th>
        <td><input type="text" name="txtPaterno"  size="80"
                   value="<?php echo $cliente[2]; ?>" /></td>
        <td></td>
      </tr>
      <tr>
        <th>MATERNO</th>
        <td><input type="text" name="txtMaterno" size="80"
                   value="<?php echo $cliente[3]; ?>" /></td>
        <td></td>
      </tr>
      <tr>
        <th>DIRECCION</th>
        <td><input type="text" name="txtDireccion"  size="80"
                   value="<?php echo $cliente[4]; ?>" /></td>
        <td></td>
      </tr>
      <tr>
        <th>TELEFONO</th>
        <td><input type="text" name="txtTelefono"  size="80"
                   value="<?php echo $cliente[5]; ?>" /></td>
        <td></td>
      </tr>
      <tr>
        <th>DISTRITO</th>
        <td><select name="selDistrito">
              <?php
                 //Llenando los distritos en el cuadro combinado
                 $seleccionado='';
                 while($distritos=mysql_fetch_array($rsD)){
                    if ($distritos[1]==$cliente[6])
                       $seleccionado=' SELECTED ';
                       else
                       $seleccionado ='';
               ?>
               <option value="<?php echo $distritos[0]; ?>"
                              <?php echo $seleccionado; ?>>
                              <?php echo $distritos[1]; ?></option>
                 <?php
                   }
                 ?>
             </select>
        </td>
        <td id="centrado"><input type="submit" name="btnActualiza"
                                 value="ACTUALIZAR" /></td>
       </tr>
       <tr>
         <th>CORREO</th>
         <td><input type="text" name="txtCorreo"  size="80"
                    value="<?php echo $cliente[7]; ?>" /></td>
```

```
            <td id="centrado"><a href="listado.php">
                                        Retornar al listado</a>
            </td>
          </tr>
        </table>
      </form>
    </section>
    <footer>
      <?php include('pie.php'); ?>
    </footer>
  </body>
</html>
```

Archivo: **grabaActualizacion.php**

```
<?php
    //Obtenemos las variables de conexión desde conecta.php
    include('conecta.php');

    //Capturamos los valores ingresados en el formulario
    $codigo=$_POST['txtCodigo'];
    $nombres=$_POST['txtNombres'];
    $paterno=$_POST['txtPaterno'];
    $materno=$_POST['txtMaterno'];
    $direccion=$_POST['txtDireccion'];
    $telefono=$_POST['txtTelefono'];
    $distrito=$_POST['selDistrito'];
    $correo=$_POST['txtCorreo'];

    //Ejecutamos el procedimiento que permite la actualización
    //del cliente, para lo cual se envian todos los valores capturados.
    $rs=  mysql_query("CALL sp_actualizacliente('$codigo','$nombres',
                                                '$paterno','$materno',
                                                '$direccion','$telefono',
                                               '$distrito','$correo')",$cn);

    //Dependiendo de la respuesta emitida por la ejecución del procedimiento
    //almacenado, emitiremos un mensaje al usuario (1) Ok-(0) Error
    if ($rs==1){
      echo "<script>
              alert('Cliente actualizado correctamente');
              location.href='listado.php';
            </script>";
    }
    else{
      echo '<script>
              alert ("Error al actualizar al cliente-'.mysql_error().'");
              location.href="listado.php";
            </script>';
    }
?>
```

Archivo: **elimina.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
```

```
    <title></title>
    <link href="estilo.css" rel="stylesheet">
  </head>
  <body>
    <header>
      <?php include('encabezado.php'); ?>
      <h3>ELIMINACION DE DATOS DE CLIENTE</h3>
    </header>
    <section>
      <?php
        include('conecta.php');
        $codigo=$_GET['id'];

        $rs=mysql_query("CALL sp_busca_cliente('$codigo')",$cn);
        $cliente=  mysql_fetch_array($rs);
      ?>
      <form method="POST" action="grabaEliminacion.php">
        <table border="1" width="900" cellspacing="1" cellpadding="1">
          <tr>
           <th>CODIGO</th>
           <td><input type="hidden" name="txtCodigo"
                      value="<?php echo $cliente[0]; ?>" />
                           <?php echo $cliente[0]; ?></td>
           <td rowspan="6">
             <?php
               //Mostrando la imagen del cliente
               echo "<img src=fotos/".$cliente[0].".jpg width=90>";
             ?>
           </td>
          </tr>
          <tr>
            <th>NOMBRES</th>
            <td><?php echo $cliente[1]; ?></td>
            <td></td>
          </tr>
          <tr>
            <th>PATERNO</th>
            <td><?php echo $cliente[2]; ?></td>
            <td></td>
          </tr>
          <tr>
            <th>MATERNO</th>
            <td><?php echo $cliente[3]; ?></td>
            <td></td>
          </tr>
          <tr>
            <th>DIRECCION</th>
            <td><?php echo $cliente[4]; ?></td>
            <td></td>
          </tr>
          <tr>
            <th>TELEFONO</th>
            <td><?php echo $cliente[5]; ?></td>
            <td></td>
          </tr>
          <tr>
            <th>DISTRITO</th>
            <td><?php echo $cliente[6]; ?></td>
            <td id="centrado"><input type="submit"
                name="btnElimina" value="ELIMINAR" /></td>
          </tr>
```

```
            <tr>
              <th>CORREO</th>
              <td><?php echo $cliente[7]; ?></td>
              <td id="centrado"><a href="listado.php">
                                  Retornar al listado</a></td>
            </tr>
          </table>
        </form>
      </section>
      <footer>
        <?php include('pie.php'); ?>
      </footer>
    </body>
  </html>
```

Archivo: **grabaEliminacion.php**

```
<?php
  include('conecta.php');
  $codigo=$_POST['txtCodigo'];

  $rs=  mysql_query("CALL sp_elimina_cliente('$codigo')",$cn);
  if ($rs==1){
    echo "<script>
            alert('Cliente eliminado correctamente');
            location.href='listado.php';
          </script>";
  }
  else{
    echo '<script>
            alert ("Error al eliminar al cliente - '.mysql_error().'");
            location.href="listado.php";
          </script>';
  }
?>
```

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
#centrado{
    text-align: center;
}
table {
    margin: auto;
}
img{
    margin: auto;
    display: block;
}
td {
    border: solid 1px #006699;
    border-top: #006699;
    border-right: #006699;
    border-left: #006699;
    border-bottom: #006699;
}
```

CAP.

4

Reportes

4.1 Clase FPDF()

A. Definición

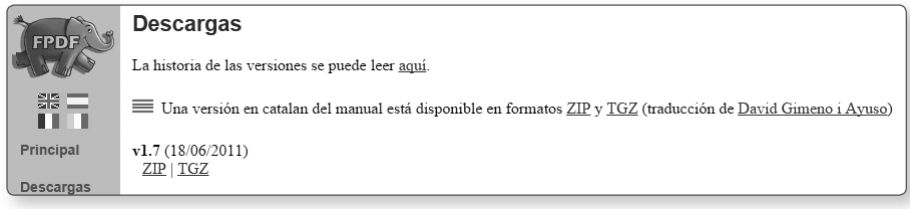
FPDF es una clase implementada con los métodos necesarios para generar todo tipo de reportes en PDF, una de sus características es controlar la propia hoja de impresión y los elementos que se distribuirán por el documento.

Esto ayudará en la generación de informes, catálogos y resúmenes; integrando así las tablas de la base de datos de MySQL con el lenguaje PHP.

B. Descarga

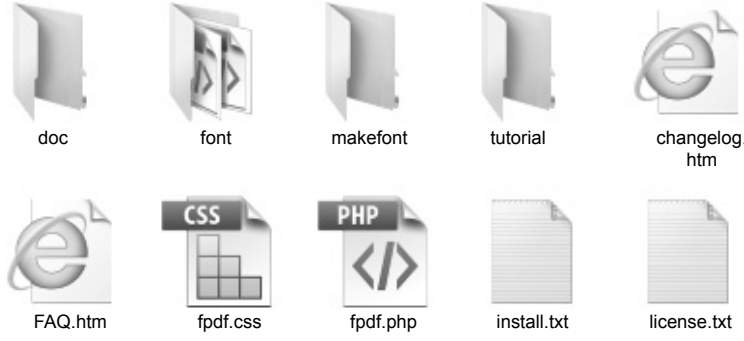
Ingrese a la siguiente URL: **http://www.fpdf.org/**

Desde aquí seleccione la opción **Descargas** para descargar la librería FPDF.



La versión más actualizada hasta la edición de este material es v1.7; para poder descargarla seleccione el enlace **ZIP**.

El contenido de la descarga es:



De todos los archivos recabados de la descarga, nos concentraremos en la carpeta **font** que contiene archivos con las fuentes que pueden ser usadas en el reporte PDF; también el archivo **fpdf.php** que contiene las funciones que permitirán implementar un reporte administrado totalmente por el usuario.

C. Instalación

La librería FPDF no necesita instalarse, lo que se debe realizar es copiar la carpeta **font** y el archivo **fpdf.php** en la carpeta de su proyecto, de forma que pueda ser invocado por cualquier otro archivo php con la función *require*.

font encabezado.php fpdf.php

En la imagen se presenta el archivo **encabezado.php** que permite implementar un reporte PDF usando la librería FPDF y la carpeta **font**.

4.2 Principales funciones de la librería FPDF

4.2.1 Objeto FPDF

A. Definición

Para construir objetos de la clase FPDF se debe tener el archivo **fpdf.php** en nuestro proyecto.

B. Sintaxis para crear un objeto de la clase FPDF

```
FPDF(orientación, unidad, formato);
```

- orientación**:** Aquí se define la forma de diseñar la página del reporte, es decir, de forma vertical o apaisada.

  - La forma vertical es la definición estándar de la hoja, para especificarlo se debe colocar el parámetro **P**.

  - La forma apaisada u horizontal se define con el parámetro **L**.

- unidad: Es la medida de usuario y sus posibles valores son pt (punto), mm (milímetro), cm (centímetro) e in (pulgada). El valor por defecto es mm.
- formato de la página: Puede tener los valores A3, A4, A5, Letter y Legal. El valor por defecto es A4.

Ejemplo: Implementar el documento PDF de orientación horizontal, con la unidad de medida en puntos y formato de hoja A4.

```
<?php
     require('fpdf.php');
    $pdf=new FPDF('L','pt','A4');
?>
```

4.2.2 Añadir una nueva página PDF

A. Definición

Esta función añade una página nueva al documento PDF. Como parámetros tan solo tiene la orientación y el formato, el resto de características se toma por defecto del método constructor.

B. Sintaxis para crear un objeto de la clase FPDF

```
AddPage(orientación, formato);
```

- orientación: Determina la orientación de la hoja mostrada en el documento PDF. Si dicha especificación fue realizada al crear el objeto de la clase FPDF, ya no será necesario volverla a especificar.
- formato: Aquí se especifica el formato de la hoja del documento PDF.

Los parámetros presentados en la función AddPage no serán necesarios especificarlos, ya que al momento de crear el objeto de la clase FDPF fueron especificados, por lo tanto, solo será necesario el formato AddPage().

Ejemplo: Implementar el documento FPDF de orientación vertical, con la unidad de medida en puntos y formato de hoja A4.

```
<?php
            require('fpdf.php');
    $pdf=new FPDF('L','pt','A4');
    $pdf->AddPage();
?>
```

4.2.3 Definición del tipo de letra y tamaño del texto

A. Comentario

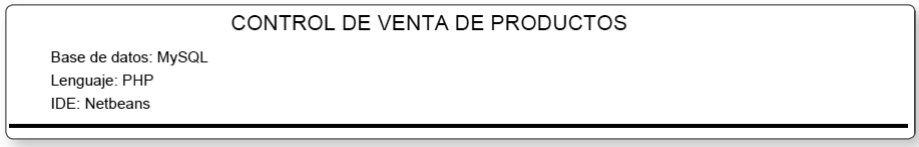
Esta función permite establecer el formato de la fuente utilizada en los textos que se mostrarán en el informe del archivo PDF. Es obligatorio llamar a esta función al inicio de la creación del archivo, de lo contrario el documento no sería válido.

B. Sintaxis

```
SetFont(familia, estilo, size);
```

- familia**:** Las familias aceptadas como valores permitidos son Courier, Helvetica, Arial, Times, Symbol, ZapfDingbats.
- estilo**:** Los estilos pueden ser regular "" (dejar vacío), negrita "B", itálica "I" y subíndice "U".
- size**:** Tamaño de la fuente en puntos. Su valor por defecto es 12 pt.

Caso 1: Implementar una aplicación en PHP que permita mostrar un encabezado textual en un documento PDF, tal como se muestra en la siguiente imagen:



Archivo: **encabezado.php**

```
<?php
    require('fpdf.php');

    //Configurando el PDF
    $pdf=new FPDF('P','pt','A4');

    //Añadiendo una nueva página
    $pdf->AddPage();

    //Definición del tipo de letra y tamaño para el titulo
    $pdf->SetFont('helvetica','',14);
    $pdf->Cell(500,30,"CONTROL DE VENTA DE PRODUCTOS",0,2,"C");

    //Definición del tipo de letra y tamaño de los textos
    $pdf->SetFont('helvetica','',10);
    $pdf->Cell(500,15,"Base de datos: MySQL",0,2,"L");
    $pdf->Cell(500,15,"Lenguaje: PHP",0,2,"L");
    $pdf->Cell(500,15,"IDE: Netbeans",0,2,"L");

    //Definición del color de la línea
    $pdf->SetDrawColor(0, 0, 0);

    //Definición del grosor de línea
    $pdf->SetLineWidth(2);
```

```
        //Implementación de la línea
        $pdf->Line(5,110,590,110);

        //Envió al documento para su visualización
        $pdf->Output('listado.pdf',"d");
    ?>
```

4.2.4 Definición de una celda para un elemento textual

A. Comentario

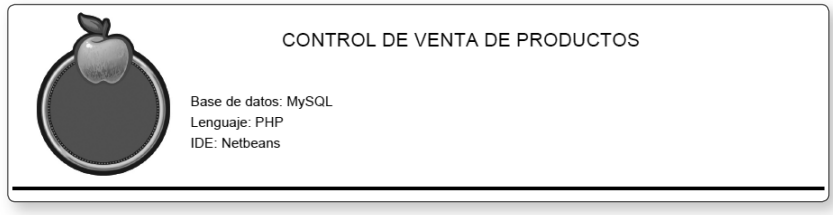
Esta función implementa una celda donde se va a imprimir un elemento de tipo texto. Tiene bordes opcionales y un color de fondo. En esta celda, se puede colocar el texto alineado o centrado.

B. Sintaxis

```
cell(w, h, texto, borde, ln, align, fill, link);
```

- w: Representa el ancho de la celda, el valor 0 extiende el margen del marco hacia el lado derecho ocupándolo todo.
- h: Representa el alto de la celda.
- texto: Aquí se especifica el texto que se quiere mostrar en el reporte PDF.
- borde: Aquí se especifica la visibilidad del borde de la celda. El valor 0 representa la no visibilidad de los bordes; mientras que el valor 1 permitirá visualizar los bordes del borde de la celda.
- ln: Donde se empezará a escribir después de llamar a esta función. Siendo 0 a la derecha, 1 al comienzo de la siguiente línea, 2 debajo.
- align: Para alinear el texto. L alineado a la izquierda, C centrado y R alineado a la derecha.
- fill: Indica si el fondo de la celda va con color o no, los valores son TRUE o FALSE.
- link: Se especifica un enlace URL desde un texto.

Caso 1: Implementar una aplicación en PHP que permita mostrar un encabezado textual con su logotipo en un documento PDF, tal como se muestra en la siguiente imagen:



Archivo: **encabezado.php**

```
<?php
    require('fpdf.php');

    //Configurando el PDF
    $pdf=new FPDF('P','pt','A4');

    //Añadiendo una nueva página
    $pdf->AddPage();

    $pdf->Image('logo.jpg' , 10 ,22,120,120);

    //Definición del tipo de letra y tamaño para el título
    $pdf->SetFont('helvetica','',14);
    $pdf->Cell(600,30,"CONTROL DE VENTA DE PRODUCTOS",0,2,"C");

    //Definición del tipo de letra y tamaño de los textos
    $pdf->SetFont('helvetica','',10);

    //Especificación de la posición del próximo elemento
    $pdf->SetXY(130, 80);

    //Implementación de los textos
    $pdf->Cell(400,15,"Base de datos: MySQL",0,2,"L");
    $pdf->Cell(400,15,"Lenguaje: PHP",0,2,"L");
    $pdf->Cell(400,15,"IDE: Netbeans",0,2,"L");

    //Definición del color de la línea
    $pdf->SetDrawColor(0, 0, 0);

    //Definición del grosor de línea
    $pdf->SetLineWidth(2);

    //Implementación de la línea
    $pdf->Line(5,150,590,150);

    //Envío del documento para su visualización
    $pdf->Output('listado.pdf',"d");
?>
```

4.2.5 Definición de la posición de un elemento textual

A. Comentario

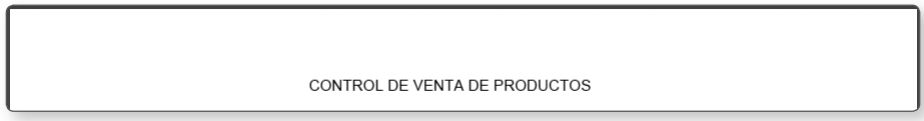
Esta función permite definir la posición del puntero en el contenido de un documento PDF, se debe tener en cuenta que su definición implica que los elementos textuales posteriores se adaptarán a dicha especificación.

B. Sintaxis

```
setXY(x,y);
```

- x: Define la posición vertical del puntero.
- y: Define la posición horizontal del puntero.

Ejemplo: Ubicar el puntero en la posición vertical 10 y horizontal 50 del documento PDF.




```
<?php
    require('fpdf.php');
    $pdf=new FPDF('P','pt','A4');
    $pdf->AddPage();

    //Definición de la posición del puntero
    $pdf->SetXY(10, 50);
    $pdf->Cell(0,0,"CONTROL DE VENTA DE PRODUCTOS",0,2,"C");

    $pdf->Output('listado.pdf',"d");
?>
```

4.2.6 Definición del color de un elemento

A. Comentario

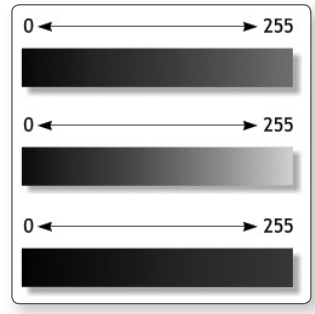
Esta función permite definir el color que pueden tener los elementos como líneas, cuadros y los bordes de las celdas. La especificación de colores usada es RGB, es decir, rojo, verde y azul. Se debe tener en cuenta que dicha definición es aplicada a todos los elementos mencionados implementados posteriormente a la definición.

B. Sintaxis

```
SetDrawColor(r,g,b)
```

- r: Define la tonalidad del color rojo, el valor permitido es de 0 a 255.
- g: Define la tonalidad del color verde, el valor permitido es de 0 a 255.
- b: Define la tonalidad del color azul, el valor permitido es de 0 a 255.

Considere la siguiente imagen para la combinación de valores RGB:



Fuente: http://www.xaraxone.com/webxealot/workbook40/edit_rgb.png

4.2.7 Definición de una línea recta

A. Comentario

Esta función permite crear una línea recta en un documento PDF.

B. Sintaxis

```
line(x1,y1, x2,y2)
```

- x1: Define la posición inicial vertical de la línea.
- y1: Define la posición inicial horizontal de la línea.
- x2: Define la posición final vertical de la línea.
- y2: Define la posición final horizontal de la línea.

Considere las ubicaciones x1,y1,x2,y2 de la siguiente manera:

Caso: Implementar una aplicación en PHP que permita crear 10 líneas rectas en un documento PDF:

Archivo: **lineas.php**

```
<?php
    require('fpdf.php');
    $pdf=new FPDF('P','pt','A4');
    $pdf->AddPage();

    //Definición del color de la línea
    $pdf->SetDrawColor(0,0,0);

    //Implementado las líneas
    for($i=1;$i<=100;$i++){
        $pdf->line(10,110+$i,580,110+$i);
        $i+=10;
    }

    $pdf->Output('listado.pdf',"d");
?>
```

4.2.8 Definición de un rectángulo

A. Comentario

Esta función permite crear un rectángulo en un documento PDF.

B. Sintaxis

```
rect(x1,y1, x2,y2)
```

- x1: Define la posición inicial vertical del rectángulo.
- y1: Define la posición inicial horizontal del rectángulo.
- x2: Define la posición final vertical del rectángulo.
- y2: Define la posición final horizontal del rectángulo.

Considere las ubicaciones x1,y1,x2,y2 de la siguiente manera:

Caso: Implementar el siguiente encabezado usando un marco de color azul para el título, tal como se muestra en la siguiente imagen:

Archivo: **encabezado.php**

```
<?php
    require('fpdf.php');
    $pdf=new FPDF('P','pt','A4');
    $pdf->AddPage();

    //Definición del color de borde
    $pdf->SetDrawColor(0,0,255);

    //Implementado un rectángulo
    $pdf->Rect(10,10,580,50);

    //Implementado el texto a mostrar
    $pdf->SetFont('helvetica','',14);
    $pdf->Cell(580,30,"CONTROL DE VENTA DE PRODUCTOS",0,2,"C");

    $pdf->Output('listado.pdf',"d");
?>
```

4.2.9 Implementación de una imagen en el documento PDF

A. Comentario

Esta función se emplea para añadir imágenes a los archivos PDF. Admite los formatos JPG, PNG y GIF.

B. Sintaxis

```
image(archivo,x,y,w,h,tipo,enlace);
```

- archivo: Especifica el nombre de la imagen a mostrar.
- x: Especifica la posición vertical de la imagen frente al documento PDF. Si no se especifica, se utilizará la posición X actual, definida por la función **setXY**.
- y: Especifica la posición horizontal de la imagen frente al documento PDF. Si no se especifica, se usará la posición Y actual, definida por la función **setXY**.
- w: Especifica el ancho que tendrá la imagen.
- h: Especifica el alto que tendrá la imagen.
- tipo: Especifica el tipo de imagen que se mostrará.
- enlace: Especifica el enlace que puede direccionar al seleccionar la imagen.

4.2.10 Definición de la salida de información

A. Comentario

Esta función permite mostrar los elementos definidos en un documento PDF, en un fichero local o en una cadena de caracteres.

B. Sintaxis

```
output(nombre,destino)
```

- nombre: Especifica el nombre que tendrá el documento PDF; si no se indica automáticamente, se le asigna el nombre por defecto **doc.pdf**.
- destino: Aquí se define el destino de envío del documento.
  - "I" envía el fichero al navegador con la opción de **Guardar como...**
  - “D” envía el documento al navegador preparado para la descarga.
  - “F” guarda el fichero en un archivo local.
  - “S” devuelve el documento como una cadena.
  - “” el documento PDF se ve directamente en el navegador.

4.2.11 Casos desarrollados

Caso desarrollado 1 **Reporte simple de clientes PDF**

Implementar una aplicación web con PHP y MySQL que permita reportar los clientes desde la base de datos Ventas2015, tal como se muestra en la siguiente imagen:

Imagen: reporteSimple.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **ReporteClientes**.
- Al iniciar la aplicación deberá mostrar el reporte directamente.
- El reporte se compone así:
  - Encabezado: Mostrar el título «Listado de Clientes» y la fecha actual de la impresión del reporte.
  - Cuerpo del reporte: Mostrar los datos del cliente como código, nombres completos, dirección, teléfono, distrito y correo electrónico.
  - Pie de página: Mostrar el total de clientes mostrados en el reporte.

El objetivo de cada archivo del proyecto es:

- **conecta.php:** Contiene las sentencias que permiten conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **fpdf**: Contiene todas las funciones de la librería fpdf.
- **reporteSimple.php**: Contiene la página web que permite mostrar el reporte PDF.

Procedimiento almacenado

```
DROP PROCEDURE IF EXISTS sp_listadoclientes;
CREATE PROCEDURE sp_listadoclientes()
    SELECT C.ID_CLIENTE,
           CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
           C.DIRECCION,C.FONO,D.DESCRIPCION,C.EMAIL
           FROM CLIENTE C
           INNER JOIN DISTRITO D
           ON C.ID_DISTRITO=D.ID_DISTRITO
           ORDER BY 1 ASC;
```

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **reporteSimple.php**

```
<?php
    include('conecta.php');
    require('fpdf.php');

    //Configurando el PDF
    $pdf=new FPDF('P');
    $pdf->AddPage();

    //Textos en el PDF
    $pdf->SetFont('Arial','B',16);
    $pdf->Cell(200,10,"LISTADO DE CLIENTES",0,1,"C");
    $pdf->Cell(10,3,"",0,1);

    //Definición de la cabecera
    $pdf->SetFont('Arial','B',8);
    $pdf->Cell(60,10,utf8_decode("Fecha de Impresión:").date("d/m/Y"),0,1);
    $pdf->Cell(60,10,"CODIGO       CLIENTE
                    DIRECCION    TELEFONO
                    DISTRITO     CORREO",0,1);

    //Impresión de los registros de la Tabla Clientes
    $pdf->SetFont('Arial','',8);

    $rs=mysql_query("CALL sp_listadoclientes()",$cn);
    while ($clientes=mysql_fetch_array($rs)){
        $codigo=$clientes[0];
        $cliente=$clientes[1];
        $direccion=$clientes[2];
        $telefono=$clientes[3];
```

```
        $distrito=$clientes[4];
        $correo=$clientes[5];

        $pdf->Cell(15,8,$codigo);
        $pdf->Cell(50,8,$cliente);
        $pdf->Cell(40,8,$direccion);
        $pdf->Cell(20,8,$telefono);
        $pdf->Cell(30,8,$distrito);
        $pdf->Cell(50,8,$correo);

        //Cambio de línea
        $pdf->Cell(50,8,'',0,1);
    }
    $pdf->Cell(50,9,"",0,1);
    $pdf->Cell(50,9,"TOTAL DE CLIENTES REGISTRADOS: "
                                        .mysql_num_rows($rs),0,1);

    //Envió al documento para su visualización
    $pdf->Output();
?>
```

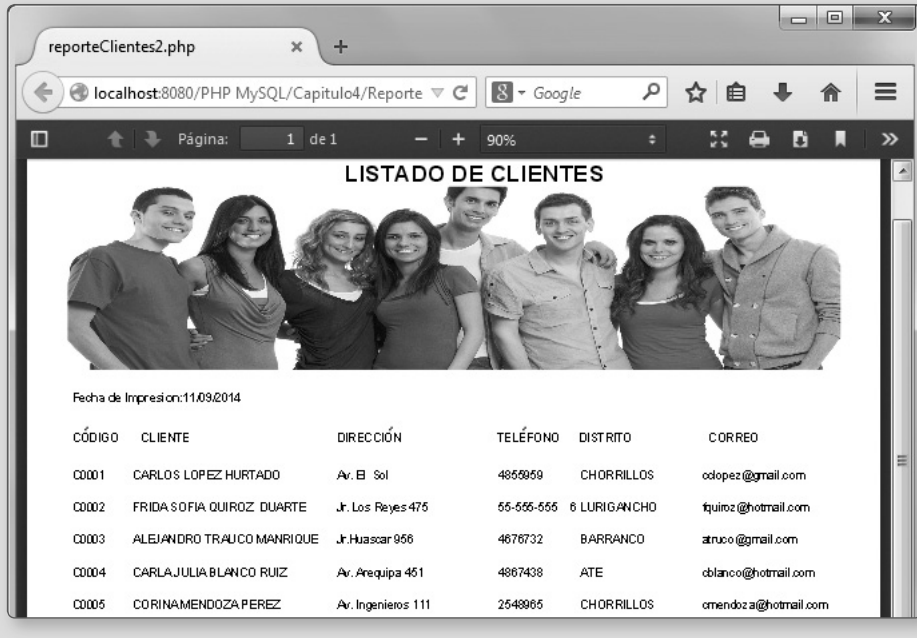
Comentarios

```
$pdf->Cell(60,10,utf8_decode("Fecha de Impresión:").date("d/m/Y"),0,1);
```

La función **utf8_decode** permite mostrar de manera correcta la letra ñ Ñ y las tildes ya que este documento no contiene etiquetas HTML, si se incluye se produce un error al generar el reporte en PDF.

Caso desarrollado 2 **Reporte de clientes con banner**

Implementar una aplicación web con PHP y MySQL que permita reportar los clientes desde la base de datos Ventas2015, tal como se muestra en la siguiente imagen:



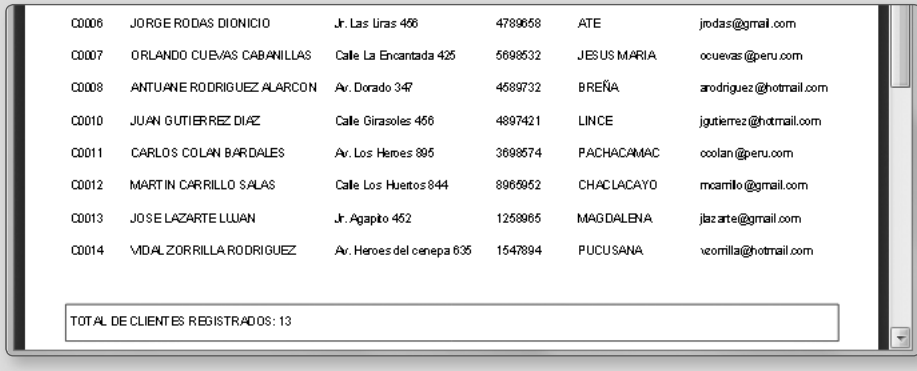


Imagen: reporteBanner.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **ReporteClientes**.
- Al iniciar la aplicación deberá mostrar el reporte directamente.
- El reporte se compone así:
  - Encabezado: Mostrar el título «Listado de Clientes», el banner de los clientes desde el archivo **banner.jpg** y la fecha actual de la impresión del reporte.
  - Cuerpo del reporte: Mostrar los datos del cliente como código, nombres completos, dirección, teléfono, distrito y correo electrónico.
  - Pie de página: Mostrar el total de clientes mostrados en el reporte.

El objetivo de cada archivo del proyecto es:

- **conecta.php**: Contiene las sentencias que permite conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **fpdf**: Contiene todas las funciones de la librería fpdf.
- **reporteBanner.php**: Contiene la página web que permite mostrar el reporte PDF.

Procedimiento almacenado

```
DROP PROCEDURE IF EXISTS sp_listadoclientes;
CREATE PROCEDURE sp_listadoclientes()
    SELECT C.ID_CLIENTE,
           CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
           C.DIRECCION,C.FONO,D.DESCRIPCION,C.EMAIL
           FROM CLIENTE C
           INNER JOIN DISTRITO D
           ON C.ID_DISTRITO=D.ID_DISTRITO
           ORDER BY 1 ASC;
```

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **reporteBanner.php**

```
<?php
    include('conecta.php');
    require('fpdf.php');

    //Configurando el PDF
    $pdf=new FPDF('P');
    $pdf->AddPage();

    //Mostrando la imagen en el reporte
    $pdf->Image('banner.jpg' ,10,18,190,45);

    //Textos en el PDF
    $pdf->SetFont('Arial','B',16);
    $pdf->Cell(200,10,"LISTADO DE CLIENTES",0,1,"C");
    $pdf->Cell(10,45,"",0,1);

    //Definición de la cabecera
    $pdf->SetFont('Arial','B',8);
    $pdf->Cell(60,10,utf8_decode("Fecha de Impresión:").date("d/m/Y"),0,1);
    $pdf->Cell(60,10,"CODIGO    CLIENTE   DIRECCION
                      TELEFONO  DISTRITO  CORREO",0,1);

    //Impresión de los registros de la Tabla Clientes
    $pdf->SetFont('Arial','',8);

    $rs=mysql_query("CALL sp_listadoclientes()",$cn);
    while ($clientes=mysql_fetch_array($rs)){
        $codigo=$clientes[0];
        $cliente=$clientes[1];
        $direccion=$clientes[2];
        $telefono=$clientes[3];
        $distrito=$clientes[4];
        $correo=$clientes[5];

        $pdf->Cell(15,8,$codigo);
        $pdf->Cell(50,8,$cliente);
        $pdf->Cell(40,8,$direccion);
        $pdf->Cell(20,8,$telefono);
        $pdf->Cell(30,8,$distrito);
        $pdf->Cell(50,8,$correo);

        //Cambio de línea
        $pdf->Cell(50,8,'',0,1);
    }

    $pdf->Cell(50,9,"",0,1);
    $pdf->Cell(190,9,"TOTAL DE CLIENTES REGISTRADOS: "
                                          .mysql_num_rows($rs),1,1);

    //Envió al documento para su visualización
    $pdf->Output();
?>
```

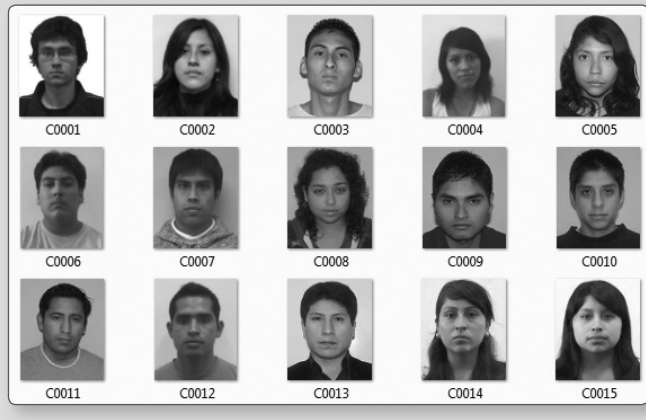
Caso desarrollado 3 **Reporte de clientes con imágenes**

Implementar una aplicación web con PHP y MySQL que permita reportar los clientes con sus respectivas imágenes desde la base de datos Ventas2015, tal como se muestra en la siguiente imagen:

Imagen: reporteImagenes.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **ReporteClientes**.
- Al iniciar la aplicación deberá mostrar el reporte directamente.
- Se debe contar con la carpeta **Fotos** con las imágenes de los clientes de la siguiente manera:



- El reporte se compone así:
  - Encabezado: Mostrar el título «Listado de Clientes», banner de clientes y la fecha actual de la impresión del reporte.
  - Cuerpo del reporte: Mostrar los datos del cliente como código, nombres completos, dirección, teléfono, distrito, correo electrónico y su imagen.
  - Pie de página: Mostrar el total de clientes mostrados en el reporte.

El objetivo de cada archivo del proyecto es:

- **conecta.php**: Contiene las sentencias que permite conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **fpdf**: Contiene todas las funciones de la librería fpdf.
- **reporteImagenes.php**: Contiene la página web que permite mostrar el reporte PDF.

Procedimiento almacenado

```
DROP PROCEDURE IF EXISTS sp_listadoclientes;
CREATE PROCEDURE sp_listadoclientes()
    SELECT C.ID_CLIENTE,
           CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
           C.DIRECCION,C.FONO,D.DESCRIPCION,C.EMAIL
           FROM CLIENTE C
           INNER JOIN DISTRITO D
           ON C.ID_DISTRITO=D.ID_DISTRITO
           ORDER BY 1 ASC;
```

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **reporteImagenes.php**

```
<?php
    include('conecta.php');
    require('fpdf.php');

    //Configurando el PDF
    $pdf=new FPDF('L');
    $pdf->AddPage();

    $pdf->Image('banner.jpg',10,22,240,60);

    //Textos en el PDF
    $pdf->SetFont('Arial','B',18);
    $pdf->Cell(220,10,"LISTADO DE CLIENTES",0,1,"C");
    $pdf->Cell(10,65,"",0,1);

    $pdf->SetFont('Arial','B',8);
    $pdf->Cell(60,8,"Fecha de Impresión:    ".
```

```
                    date("d/m/Y"),0,1);
    $pdf->Cell(60,8,"CÓDIGO CLIENTE DIRECCION    TELEFONO  DISTRITO   CORREO
                   FOTO",0,1);

    //Impresión de los registros de la Tabla Clientes
    $pdf->SetFont('Arial','',8);
    $rs=mysql_query("CALL sp_listadoclientes()",$cn);

    while ($clientes=mysql_fetch_array($rs)){
        $codigo = $clientes[0];
        $cliente = $clientes[1];
        $direccion = $clientes[2];
        $telefono = $clientes[3];
        $distrito = $clientes[4];
        $correo = $clientes[5];

        $foto='fotos/'.$codigo.'.jpg';

        //Cambio de línea
        $pdf->Cell(50,10,'',0,1);

        $pdf->Cell(20,10,$codigo);
        $pdf->Cell(60,10,$cliente);
        $pdf->Cell(50,10,$direccion);
        $pdf->Cell(25,10,$telefono);
        $pdf->Cell(30,10,$distrito);
        $pdf->Cell(40,10,$correo);
        $pdf->Image($foto,$pdf->GetX(),$pdf->GetY(),15);
        $pdf->Cell(50,10,'',0,1);
    }

    $pdf->Cell(50,8,"Total de Clientes registrados: "
                                                .mysql_num_rows($rs),0,1);

    //Envió al documento para su visualización
    $pdf->Output();
?>
```

Comentarios

```
$foto='fotos/'.$codigo.'.jpg';
```

La variable **$foto** almacena la ruta de la imagen a mostrar, se debe tener en cuenta que las imágenes se encuentran en la carpeta **Fotos**, por este motivo, se debe especificarlo antes del propio nombre de la imagen. Respecto al nombre de la imagen, esta proviene del código del cliente, por lo tanto, desde la consulta se obtiene dicho valor mediante la variable **$codigo**. Es obligatorio la especificación de la extensión de la imagen, en nuestro caso se usa imágenes de tipo JPG.

```
$pdf->Image($foto,$pdf->GetX(),$pdf->GetY(),15);
```

La variable **$pdf** permite componer todo el reporte al ejecutar la aplicación, en ese sentido la imagen también debe ser considerada dentro de la variable usando la función Image y especificando la ruta de la imagen($foto), seguidamente precisar la posición de la imagen en el reporte usando las funciones GetX y GetY, esto se debe a que la imagen es el último elemento mostrado de un cliente en el reporte, el valor 15 representa el tamaño del marco de la imagen; no se puede controlar el alto y ni ancho, solo la proporción de su marco.

Caso desarrollado 4 **Generación de fotocheck**

Implementar una aplicación web con PHP y MySQL que permita generar el fotocheck de un determinado cliente, para lo cual primero se debe buscar al cliente y mediante un enlace se podrá generar un fotocheck, mostrando el código, apellidos, nombres y su imagen; además de imprimir la fecha de generación desde la base de datos Ventas2015, tal como se muestra en la siguiente imagen:

Imagen: busqueda.php

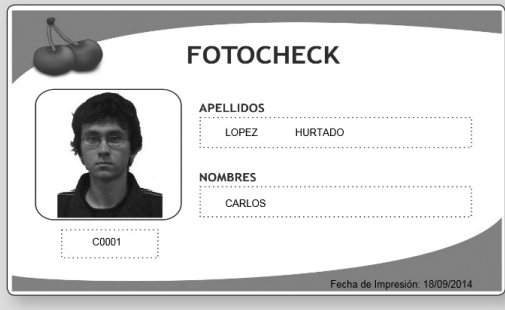


Imagen: fotocheck.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **FotocheckCliente**.
- Al iniciar la aplicación deberá mostrar la página **busqueda.php**.
- Para la generación del fotocheck se necesita:

- Un archivo de tipo JPG que muestra la imagen del fotocheck vacío, de forma que se imprimirá los valores encontrados en las posiciones respectivas de la imagen, tal como se muestra en la siguiente imagen:

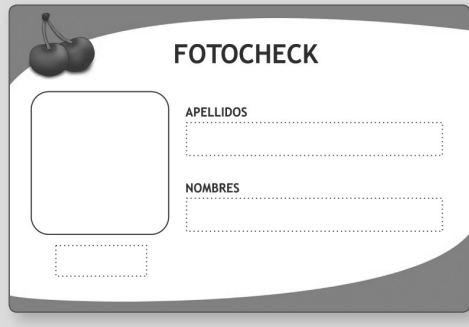


Imagen: fotocheck.jpg

Este archivo de imagen debe encontrarse en la carpeta del proyecto.

- Para asignar la imagen de un determinado cliente, se debe tener en el proyecto las imágenes de los clientes.
- Finalmente, la librería fpdf también debe encontrarse dentro de la carpeta del proyecto. En todos los casos, los archivos deben registrarse en la carpeta del servidor WAMP o XAMPP.

El objetivo de cada archivo del proyecto es:

- **conecta.php**: Contiene las sentencias que permiten conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **fpdf**: Contiene todas las funciones de la librería fpdf.
- **busqueda.php**: Contiene la página web que interactúa con el usuario, el cual permitirá buscar un determinado cliente mediante el código del mismo y mostrar sus datos personales; también, contiene un enlace que permite mostrar su fotocheck.
- **fotocheck.php**: Contiene la página que genera el fotocheck del cliente.
- **miEstilo.css**: Contiene los estilos usados en el archivo **busqueda.php**.

Procedimiento almacenado

```
DROP PROCEDURE IF EXISTS sp_busca_cliente;
CREATE PROCEDURE sp_busca_cliente(cod char(5))
    SELECT * FROM CLIENTE
                    WHERE ID_CLIENTE=COD;
```

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **busqueda.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">
  </head>
  <body>
    <header>
      <h3>FOTOCHECK DE CLIENTES</h3>
      <img src="banner.jpg" width="550" height="120"/>
    </header>
    <section>
      <?php
        error_reporting(0);
        include('conecta.php');
        include('captura.php');

        $codigo = getCodigo();

        //Determina qué botón ha seleccionado el usuario
        if (isset($_POST['btnGenerico'])) {
           $boton = $_POST['btnGenerico'];

           if ($boton == 'BUSCAR') {
             $rs = mysql_query("CALL sp_buscacliente('$codigo');", $cn);
             $cliente = mysql_fetch_array($rs);
           }
       }
      ?>
   <form method="POST">
      <table border="0" width="550" cellspacing="1" cellpadding="10">
         <tr>
            <td>Ingrese codigo</td>
            <td><input type="text" name="txtCodigo" maxlength="5"
                       value="<?php echo getCodigo(); ?>" /></td>

            <td><input type="submit" name="btnGenerico"
                       value="BUSCAR" /></td>
         </tr>
       </table>
       <br>
       <table border="0" width="550" cellspacing="1" cellpadding="5">
          <tr>
            <th>NOMBRES</th>
            <td colspan="3"><?php echo $cliente[1] ?></td>
          </tr>
          <tr>
            <th>PATERNO</th>
            <td><?php echo $cliente[2] ?></td>
            <th>MATERNO</th>
            <td><?php echo $cliente[3] ?></td>
          </tr>
          <tr>
            <th>DIRECCION</th>
            <td colspan="3"><?php echo $cliente[4] ?></td>
          </tr>
          <tr>
             <th>TELEFONO</th>
             <td><?php echo $cliente[5] ?></td>
```

```
            <th>DISTRITO</th>
            <td><?php echo $cliente[6] ?></td>
          </tr>
          <tr>
            <th>EMAIL</th>
            <td colspan="3"><?php echo $cliente[7] ?></td>
          </tr>
          <tr>
            <td colspan="4" id="centrado">
              <a href="fotocheck.php?id=<?php echo $cliente[0];?>">
                    Generar reporte
              </a>
            </td>
          </tr>
        </table>
      </form>
    </section>
    <footer>
       <h5>Todos los derechos reservados @2015</h5>
    </footer>
</body>
</html>
```

Archivo: **fotocheck.php**

```
<?php
    //Obteniendo el código de la página fotocheck.php
    $cod = $_GET['id'];

    include('conecta.php');
    require('fpdf.php');

    //Configurando el PDF
    $pdf=new FPDF('L');
    $pdf->AddPage();

    //Mostrando la imagen del fotocheck.jpg
    $pdf->Image('fotocheck.jpg',80,22,140,80);

    //Consultando al cliente de acuerdo al código obtenido
    $rs=mysql_query("SELECT * FROM CLIENTE WHERE ID_CLIENTE='$cod'",$cn);

    $pdf->SetFont('Arial','',8);

    //Transfiriendo el archivo encontrado a la variable $cliente
    while ($cliente=mysql_fetch_array($rs)){
        $codigo = $cliente[0];
        $nombres = $cliente[1];
        $paterno = $cliente[2];
        $materno = $cliente[3];
        $foto='fotos/'.$codigo.'.jpg';

        $pdf->SetXY(102, 78);
        $pdf->Cell(20,20,$codigo);

        $pdf->SetXY(93, 45);
        $pdf->Image($foto,$pdf->GetX(),$pdf->GetY(),30);
```

```
            $pdf->SetXY(140, 47);
            $pdf->Cell(20,20,$paterno);
            $pdf->Cell(20,20,$materno);

            $pdf->SetXY(140, 67);
            $pdf->Cell(20,20,$nombres);
        }

        //Impresión de fecha
        $pdf->SetXY(170, 95);
        $pdf->Cell(160,10,utf8_decode("Fecha de Impresión: ").date("d/m/Y"));
        $pdf->Output();
    ?>
```

Comentarios

```
$pdf->SetXY(102, 78);
```

La función SetXY permite ubicar el puntero dentro del documento en una determinada posición X, Y. Donde 102 representa el valor X y 78 al valor de Y.

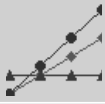
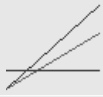
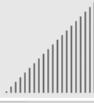
Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
table {
    margin: auto;
    background-color: antiquewhite;
}
img{
    margin: auto;
    display: block;
}
table th {
    font-size: 10px;
    background-color: #b9c9fe;
}
#centrado{
    text-align: center;
    font-size: 18px;
}
```

4.3 Clase PHPLOT

Es una librería que permite crear archivos de tipo estadísticos; mostrando imágenes científicas, de negocio y otros tipos de gráficos. La implementación de los gráficos se realiza por medio de una aplicación PHP; incluyendo la librería tendremos acceso a todos los métodos que permiten generar gráficos de tipo pie, bar, line, etc.

4.3.1 Tipos de gráficos

	AREA: permite rellenar áreas entre las líneas.
	BARS: permite rellenar las barras de forma vertical, opcionalmente se puede presentar en 3D usando el método SetPlotAreaWorld.
	BOXES: permite mostrar un resumen estadístico de números.
	BUBBLES: permite mostrar círculos rellenos de tamaños referentes al valor mostrado.
	LINEPOINTS: permite mostrar puntos entre líneas, indicando los valores referentes al valor mostrado.
	LINES: permite mostrar puntos entre líneas, indicando los valores referentes al valor mostrado.
	PIE: permite mostrar un bloque circular tipo pastel, indicando los valores referentes al valor mostrado. Opcionalmente se puede mostrar en 3D.
	THINBARLINE: muestra líneas, indicando los valores referentes al valor mostrado.

4.3.2 Caso desarrollado: Generación de gráficos estadísticos

Implementar una aplicación web con PHP y MySQL que permita generar un gráfico estadístico del total de clientes por distrito, tal como se muestra en la siguiente imagen:

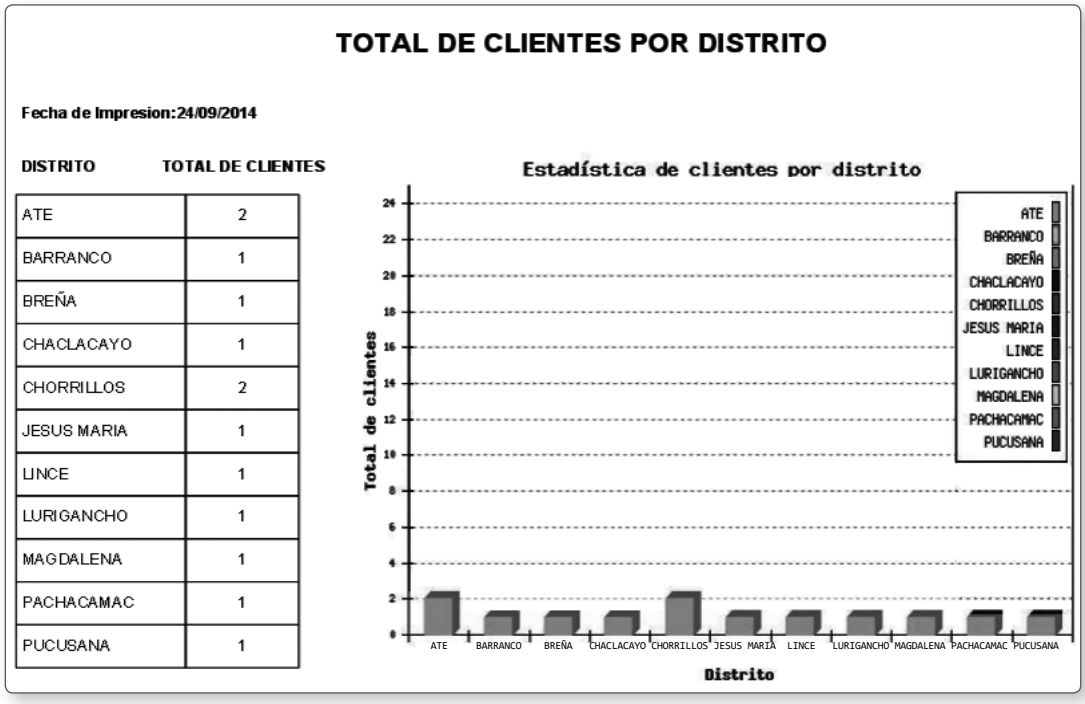


Imagen: reporteClientesxDistrito.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **GraficosPDF**.
- Para iniciar la aplicación se debe ejecutar el archivo **clientesxdistrito.php** que tiene por misión generar el archivo **grafico.jpg**, el cual contendrá la imagen estadística de la consulta.
- Para generar el gráfico estadístico se necesita:
  - La librería phplot provee de métodos que permiten realizar el gráfico estadístico, este archivo debe ser incorporado en la carpeta del proyecto.
  - La librería fpdf provee de métodos que permiten realizar la implementación del documento PDF, este archivo debe ser incorporado en la carpeta del proyecto.

El objetivo de cada archivo del proyecto es:

- **conecta.php:** Contiene las sentencias que permiten conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **fpdf:** Contiene todas las funciones de la librería fpdf.
- **fpdf:** Contiene todas las funciones de la librería fpdf.
- **clientesxdistrito.php:** Contiene la página web que interactúa con el usuario, la cual permitirá buscar un determinado cliente mediante el código del mismo y mostrará sus datos personales; también, contiene un enlace que permite mostrar su fotocheck.
- **reporteClientesxDistrito.php:** Contiene la página que genera el fotocheck del cliente.

Procedimiento almacenado

```
DROP PROCEDURE IF EXISTS sp_clientesxdistrito;
CREATE PROCEDURE sp_clientesxdistrito()
    SELECT D.DESCRIPCION,COUNT(*)
            FROM CLIENTE C
            INNER JOIN DISTRITO D ON C.ID_DISTRITO=D.ID_DISTRITO
            GROUP BY D.DESCRIPCION;
```

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **clientesxdistrito.php**

```
<?php
    //Incluyendo las librerías de conexión y de la imagen estadística
    include('conecta.php');
    include('phplot.php');

    //Instancia de la clase PHPlot
    $gra=new PHPlot();
    $gra->SetPrintImage(false);
    $gra->SetFileFormat("jpg");
    $gra->SetOutputFile("grafico.jpg");
    $gra->SetIsInline(true);
    $gra->SetDataType("text-data");

    //Ejecutando la consulta de agrupamiento de clientes por distrito
    $rs=mysql_query("CALL sp_clientesxdistrito()", $cn);

    //Llenando los arreglos para mostrarlo en el gráfico
    while ($clientes = mysql_fetch_array($rs)) {
        $data[]=array(utf8_decode($clientes[0]),$clientes[1]);
        $cols[]=utf8_decode($clientes[0]);
    }

    //Implementando el gráfico con los datos del arreglo
    $gra->SetDataValues($data);
    $gra->SetLineStyles("solid");

    //Selección de tipo de gráfico
    $gra->SetPlotType("bars");

    //Título de gráfico estadístico
    $gra->SetTitle(utf8_decode("Estadística de clientes por distrito"));

    //Título del eje X
    $gra->SetXTitle("Distrito");

    //Título del eje Y
    $gra->SetYTitle("Total de clientes");

    //Leyenda
    $gra->SetLegend($cols);
```

```
    //Escalando la barra
    $gra->SetPlotAreaWorld(NULL,0, NULL, 25);
    $gra->DrawGraph();

    //Imprimiendo la imagen
    $gra->PrintImage();

    //Invocando al archivo que mostrará la imagen en el PDF
    header("location:reporteClientesxDistrito.php");
?>
```

Archivo: **reporteClientesxDistrito.php**

```
<?php
    include('conecta.php');
    require('fpdf.php');

    //Configurando el PDF
    $pdf=new FPDF('P');
    $pdf->AddPage();

    //Textos en el PDF
    $pdf->SetFont('Arial','B',14);
    $pdf->Cell(200,10,"TOTAL DE CLIENTES POR DISTRITO",0,1,"C");
    $pdf->Cell(10,3,"",0,1);

    //Imprimiendo el gráfico estadístico
    $pdf->Image("grafico.jpg",70,35,130,100);

    //Definición de la cabecera
    $pdf->SetFont('Arial','B',8);
    $pdf->Cell(60,10,"Fecha de Impresion:".date("d/m/Y"),0,1);
    $pdf->Cell(60,10,"DISTRITO                TOTAL DE CLIENTES",0,1);

    //Impresión de los registros de la tabla clientes
    $pdf->SetFont('Arial','',8);

    $rs=mysql_query("CALL sp_clientesxdistrito()",$cn);
    while ($consulta=mysql_fetch_array($rs)){
        $distrito=$consulta[0];
        $total=$consulta[1];

        //Imprimir la lista de distritos y el total de clientes
        $pdf->Cell(30,8,$distrito,1,0);
        $pdf->Cell(20,8,$total,1,0,"C");

        //Cambio de línea
        $pdf->Cell(50,8,'',0,1);
    }

    //Envío del documento para su visualización
    $pdf->Output();
?>
```

CAP.

5

Manejo de imágenes

5.1 Introducción

El presente capítulo tiene la misión de implementar aplicaciones PHP conectadas a la base de datos MySQL para generar gráficos de tipo estadísticos; el cual permitirá mostrar los datos en forma gráfica y así poder tomar decisiones en una determinada situación.

Además, se debe mencionar que para el desarrollo de las aplicaciones se debe contar con la librería PHPLot; el cual permitirá generar los gráficos de manera profesional.

5.1.1 Casos desarrollados

Caso desarrollado 1 **Gráfico básico-Listado de clientes por distrito**

Implementar una aplicación web con PHP y MySQL que permita listar el total de clientes registrados por distrito y, además, mostrar una barra horizontal que represente al total de clientes por distrito, tal como se muestra en la siguiente imagen:

LISTADO DE CLIENTES POR DISTRITO

DISTRITO	TOTAL DE CLIENTES	GRÁFICO
ATE	3	
BARRANCO	1	
BREÑA	2	
CARABAYLLO	1	
CERCADO	1	
CHACLACAYO	1	
JESUS MARIA	2	
LINCE	1	
MAGDALENA	1	
PACHACAMAC	2	
PUCUSANA	1	

Todos los derechos reservados @2015

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **GráficoBásico**.
- Al iniciar la aplicación deberá mostrar la página listado.php.
- Para la generación del listado se necesita:
  - Mostrar en una tabla el nombre del distrito, número de clientes por distrito y una barra horizontal que represente el número de clientes.

- Un archivo de tipo JPG que contenga un rectángulo, tal como se muestra en la siguiente imagen , la idea central del caso es aumentar el ancho de acuerdo al valor obtenido desde una consulta MySQL.

El objetivo de cada archivo que contiene el proyecto es:

- **conecta.php:** Contiene las sentencias que permiten conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **encabezado.php:** Contiene el banner y el título del documento web.
- **pie.php:** Contiene el texto «Todos los derechos reservados» que se muestra en el pie de página del documento.
- **listado.php:** Contiene la página que genera el listado de los distritos, el total de clientes por distrito y la barra que, de acuerdo a su ancho, representa el total de clientes por distrito.
- **miEstilo.css:** Contiene los estilos usados en el archivo **listado.php**.

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **encabezado.php**

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
     <img id="imagen" src="banner.jpg" width="720" height="180"/>
        <h3> LISTADO DE CLIENTES POR DISTRITO </h3>
    </body>
</html>
```

Archivo: **pie.php**

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <h5>Todos los derechos reservados @2015</h5>
    </body>
</html>
```

Archivo: **listado.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">
  </head>
  <body>
    <header>
       <?php
         include('encabezado.php');
         include('conecta.php');
       ?>
    </header>
    <section>
      <form method="POST">
        <table border="1" width="400" cellspacing="0" cellpadding="5">
          <tr>
            <th>DISTRITO</th>
            <th id="centrado">TOTAL DE CLIENTES</th>
            <th>GRAFICO</th>
          </tr>
          <?php
             $rs=mysql_query("SELECT D.DESCRIPCION,COUNT(*)
                              FROM CLIENTE C
                              INNER JOIN DISTRITO D
                              ON C.ID_DISTRITO=D.ID_DISTRITO
                              GROUP BY D.DESCRIPCION;",$cn);

             while($distritos = mysql_fetch_array($rs)){
          ?>
          <tr>
            <td><?php echo utf8_decode($distritos[0]); ?></td>
            <td id="centrado"><?php echo $distritos[1]; ?></td>
            <td id="izquierda">
               <?php $total=$distritos[1];?>
               <img src="rectangulo.jpg"
                    width="<?php echo $total*100; ?>" height="20" />
            </td>
          </tr>
          <?php } ?>
        </table>
      </form>
    </section>
    <footer>
       <?php include('pie.php') ?>
    </footer>
  </body>
</html>
```

Comentarios

```
<?php
   $rs=mysql_query("SELECT D.DESCRIPCION,COUNT(*)
                    FROM CLIENTE C
                    INNER JOIN DISTRITO D
                    ON C.ID_DISTRITO=D.ID_DISTRITO
                    GROUP BY D.DESCRIPCION;",$cn);

   while($distritos = mysql_fetch_array($rs)){
?>
```

La variable **$rs** ejecuta la consulta que permite mostrar la descripción de los distritos, además de contar cuántos clientes hay por distrito, usando para esto la unión de tabla con la cláusula **INNER JOIN** de la sentencia **SELECT** y agrupándolo por la descripción del distrito.

Luego se implementa la estructura repetitiva while que permitirá recorrer por todos los registros obtenidos en la variable **$rs**; la variable **$distritos** tiene acceso a todos los registros mediante la función **mysql_fetch_array**, también se puede especificar con la función **mysql_fetch_row**.

```
<tr>
  <td><?php echo utf8_decode($distritos[0]); ?></td>
  <td id="centrado"><?php echo $distritos[1]; ?></td>
  <td id="izquierda">
     <?php $total=$distritos[1];?>
     <img src="rectangulo.jpg"
          width="<?php echo $total*100; ?>" height="20" />
  </td>
</tr>
<?php } ?>
```

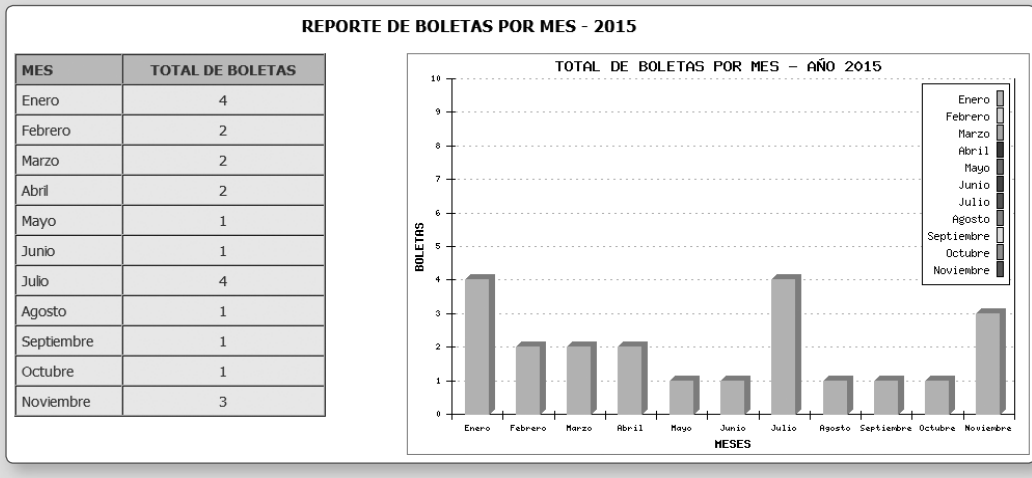
Según el trabajo realizado por la estructura repetitiva while, la variable **$distritos** se desprende en «Nombre del distrito» (**distritos[0]**) y el «Total de clientes por distrito» **$distritos[1]**, estos son impresos en sus celdas correspondientes. La variable **$total** obtiene el número de clientes por cada distrito, cada valor de la variable **$total** será impreso en el ancho (width) del archivo **rectángulo.gif** con una altura de 20px, cabe mencionar que el valor de la variable **$total** se multiplicó por 100 para que el ancho sea representativo, de otra manera los anchos serían muy cortos.

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
#centrado{
    text-align: center;
}
#izquierda{
    text-align: left;
}
table {
    margin: auto;
    background-color: antiquewhite;
}
#imagen{
    margin: auto;
    display: block;
}
table th {
    background-color: #b9c9fe;
}
```

Caso desarrollado 2 Gráfico con PhpLot-Reporte de boletas por mes-Año 2015

Implementar una aplicación web con PHP y MySQL que permita reportar el número total de boletas registradas en el año 2015, agrupadas por meses, tal como se muestra en la siguiente imagen:



MES	TOTAL DE BOLETAS
Enero	4
Febrero	2
Marzo	2
Abril	2
Mayo	1
Junio	1
Julio	4
Agosto	1
Septiembre	1
Octubre	1
Noviembre	3

Imagen: listado.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **BoletasxMes2015**.
- Descargar la librería necesaria para la implementación de los gráficos desde la página http://sourceforge.net/projects/phplot/files/. Una vez descargada la librería, copie el archivo **phplot.php** al proyecto.

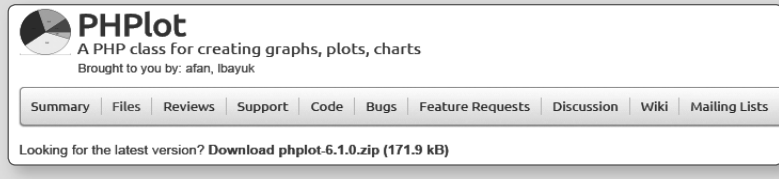


Imagen: http://sourceforge.net/projects/phplot/files/

- Al iniciar la aplicación deberá mostrar la página **listado.php**.
- Para la generación del listado se necesita:
  - Mostrar en una tabla el nombre del mes en castellano y el número de boletas por mes.
  - El gráfico deberá mostrarse en el lado derecho de la tabla, tal como se muestra en la imagen inicial de la aplicación.

El objetivo de cada archivo que contiene el proyecto es:

- **conecta.php:** Contiene las sentencias que permiten conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **phplot.php:** Contiene todas las funciones que permiten implementar gráficos.
- **grafico.php:** Contiene las sentencias necesarias para crear el gráfico de barras de forma vertical, este archivo será mostrado en **listado.php**.

- **listado.php:** Contiene la página principal que permite interactuar la aplicación con el usuario, así como mostrar la imagen y la tabla de valores.
- **miEstilo.css:** Contiene los estilos usados en el archivo **listado.php**.

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **grafico.php**

```
<?php
    include('conecta.php');
    include('phplot.php');

    //La variable $data se encarga de almacenar los datos a graficar
    $data=array();

    //Creando una instancia nueva de la clase PHPlot esto es debido a la
    //creación de una nueva imagen.
    $gra=new PHPlot();

    //Implementación de la consulta
    $sql="SELECT (CASE WHEN MONTH(FECHA) = 1 THEN 'Enero'
                       WHEN MONTH(FECHA) = 2 THEN 'Febrero'
                       WHEN MONTH(FECHA) = 3 THEN 'Marzo'
                       WHEN MONTH(FECHA) = 4 THEN 'Abril'
                       WHEN MONTH(FECHA) = 5 THEN 'Mayo'
                       WHEN MONTH(FECHA) = 6 THEN 'Junio'
                       WHEN MONTH(FECHA) = 7 THEN 'Julio'
                       WHEN MONTH(FECHA) = 8 THEN 'Agosto'
                       WHEN MONTH(FECHA) = 9 THEN 'Septiembre'
                       WHEN MONTH(FECHA) = 10 THEN 'Octubre'
                       WHEN MONTH(FECHA) = 11 THEN 'Noviembre'
                       WHEN MONTH(FECHA) = 12 THEN 'Diciembre'
                  END) AS MES, COUNT(*) AS TOTAL
                  FROM BOLETA
                  WHERE YEAR(FECHA)=2015
                  GROUP BY MONTH(FECHA);";
    $rs=mysql_query($sql, $cn);

    //Obteniendo los valores desde la consulta
    while ($row = mysql_fetch_row($rs)) {
        $data[]=array(utf8_decode($row[0]),$row[1]);
        $cols[]=utf8_decode($row[0]);
    }

    //Enviando los valores al gráfico
    $gra->SetImageBorderType('plain');
    $gra->SetDataValues($data);
    $gra->SetLineStyles("solid");
    $gra->SetPlotType("bars");

    //Especificando los títulos de la imagen
    $gra->SetTitle(utf8_decode("TOTAL DE BOLETAS POR MES - AÑO 2015"));
    $gra->SetXTitle("MESES");
    $gra->SetYTitle("BOLETAS");
    //Especificando la leyenda
    $gra->SetLegend($cols);
```

```
    //Especificando la proyección del área
    $gra->SetPlotÁreaWorld(NULL,0, NULL, 10);
    $gra->DrawGraph();
?>
```

Archivo: **listado.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">
  </head>
  <body>
    <header>
      <?php
         include('conecta.php');
      ?>
      <h3> REPORTE DE BOLETAS POR MES - 2015 </h3>
    </header>
    <section>
      <article id="columna">
        <table border="1" width="300" cellspacing="0" cellpadding="5">
          <tr>
           <th>MES</th>
           <th id="centrado">TOTAL DE BOLETAS</th>
          </tr>
          <?php
            $rs=mysql_query("SELECT (CASE WHEN MONTH(FECHA) = 1 THEN 'Enero'
                                    WHEN MONTH(FECHA) = 2 THEN 'Febrero'
                                    WHEN MONTH(FECHA) = 3 THEN 'Marzo'
                                    WHEN MONTH(FECHA) = 4 THEN 'Abril'
                                    WHEN MONTH(FECHA) = 5 THEN 'Mayo'
                                    WHEN MONTH(FECHA) = 6 THEN 'Junio'
                                    WHEN MONTH(FECHA) = 7 THEN 'Julio'
                                    WHEN MONTH(FECHA) = 8 THEN 'Agosto'
                                    WHEN MONTH(FECHA) = 9 THEN 'Septiembre'
                                    WHEN MONTH(FECHA) = 10 THEN 'Octubre'
                                    WHEN MONTH(FECHA) = 11 THEN 'Noviembre'
                                    WHEN MONTH(FECHA) = 12 THEN 'Diciembre'
                                    END) AS MES, COUNT(*) AS TOTAL
                                    FROM BOLETA
                                    WHERE YEAR(FECHA)=2015
                                    GROUP BY MONTH(FECHA);",$cn);
            while($boletas = mysql_fetch_array($rs)){
          ?>
          <tr>
           <td><?php echo utf8_decode($boletas[0]); ?></td>
           <td id="centrado"><?php echo $boletas[1]; ?></td>
          </tr>
          <?php
              }
          ?>
         </table>
        </article>
        <article id="columna">
            <img src="grafico.php" >
        </article>
      </section>
  </body>
</html>
```

Comentarios

```
<?php
   $rs=mysql_query("SELECT (CASE WHEN MONTH(FECHA) = 1 THEN 'Enero'
                                 WHEN MONTH(FECHA) = 2 THEN 'Febrero'
                                 WHEN MONTH(FECHA) = 3 THEN 'Marzo'
                                 WHEN MONTH(FECHA) = 4 THEN 'Abril'
                                 WHEN MONTH(FECHA) = 5 THEN 'Mayo'
                                 WHEN MONTH(FECHA) = 6 THEN 'Junio'
                                 WHEN MONTH(FECHA) = 7 THEN 'Julio'
                                 WHEN MONTH(FECHA) = 8 THEN 'Agosto'
                                 WHEN MONTH(FECHA) = 9 THEN 'Septiembre'
                                 WHEN MONTH(FECHA) = 10 THEN 'Octubre'
                                 WHEN MONTH(FECHA) = 11 THEN 'Noviembre'
                                 WHEN MONTH(FECHA) = 12 THEN 'Diciembre'
                                 END) AS MES, COUNT(*) AS TOTAL
                                 FROM BOLETA
                                 WHERE YEAR(FECHA)=2015
                                 GROUP BY MONTH(FECHA);",$cn);
   while($boletas = mysql_fetch_array($rs)){
?>
```

La variable **$rs** permite almacenar los registros obtenidos a partir de la consulta a la tabla Boleta, desde la cual se mostrará los meses usando la cláusula CASE WHEN que permite obtener los meses en letras desde un número de mes; obtenido con la función **MONTH**, además, agregamos la columna del conteo de boletas con la función agregada **COUNT**. No se olvide que solo se debe presentar las boletas del 2015 (WHERE YEAR(FECHA)=2015) y agrupadas por el número de meses (GROUP BY MONTH(FECHA).

Para poder recorrer todos los registros obtenidos, usaremos la estructura repetitiva **while**, esta información será registrada en la variable **$boletas** usando la función **mysql_fetch_array**.

```
<tr>
  <td><?php echo utf8_decode($boletas[0]); ?></td>
  <td id="centrado"><?php echo $boletas[1]; ?></td>
</tr>
<?php
    }
?>
```

La información obtenida en la variable **$boletas** será impresa en las celdas correspondientes; el nombre del mes se encuentra en **$boletas[0]** y el total de boletas por meses se encuentra en **$boletas[1]**.

```
<article id="columna">
   <img src="grafico.php" >
</article>
```

Para mostrar la imagen, según la consulta, se debe usar la etiqueta de HTML IMG, asignando a su propiedad SRC el archivo **grafico.php**; se debe considerar que el archivo **grafico.php** genera un archivo de tipo JPG, el cual debe ser impreso al costado de la tabla mostrada en el listado.

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
#centrado{
    text-align: center;
}
#izquierda{
    text-align: left;
}
table {
    margin: auto;
    background-color: antiquewhite;
}
img{
    margin: auto;
    display: block;
}
table th {
    background-color: #b9c9fe;
}
#columna
{
    position:relative;
    float:left;
    left:200px;
    width: 30%;
    top:0px;
    text-align:justify;
    color:#039;
    padding: 0px 5px 0px 10px;
    font-family:Tahoma, Geneva, sans-serif;
}
```

Caso desarrollado 3 Gráfico con PHPLot-Listado de boletas por cliente

Implementar una aplicación web con PHP y MySQL que permita reportar el número total de boletas registradas por cliente, además, el usuario podrá seleccionar varios tipos de gráficos como barras verticales, pastel, líneas, barras horizontales y puntos de dispersión, tal como se muestra en la siguiente imagen:

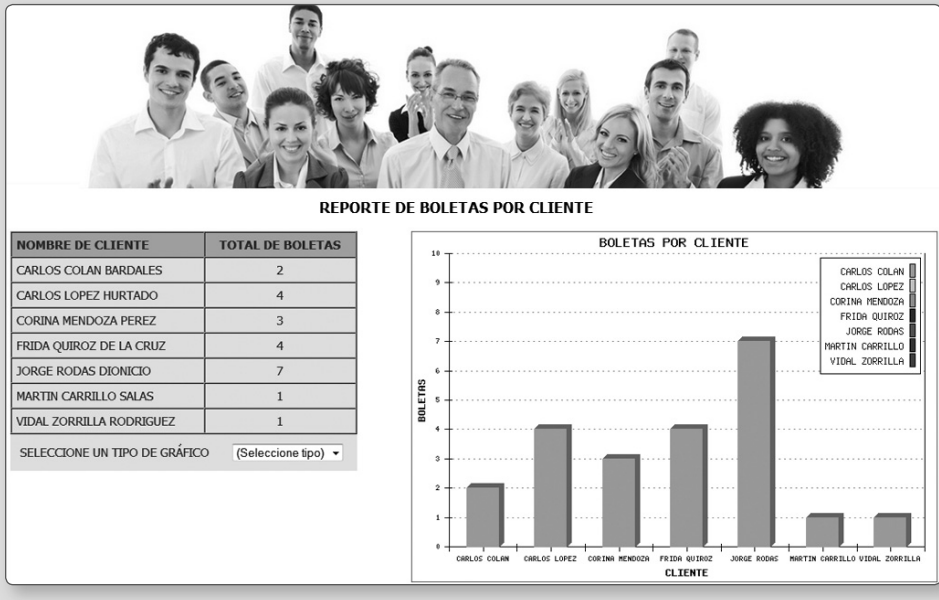
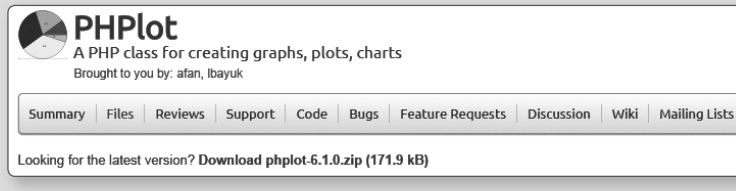


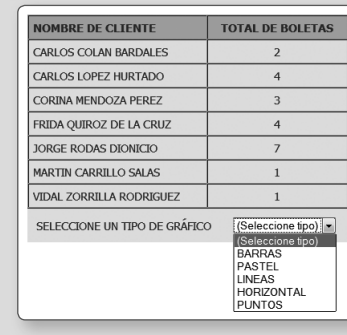
Imagen: listado.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **BoletasxCliente**.
- Al iniciar la aplicación deberá mostrar la página **listado.php**.
- Descargar la librería necesaria para la implementación de los gráficos desde la página http://sourceforge.net/projects/phplot/files/. Una vez descargada la librería copie el archivo **phplot.php** al proyecto.



- Para la generación del listado se necesita:
  - Mostrar en una tabla el nombre completo del cliente y el número de boletas por cliente.
  - Además deberá mostrar un cuadro combinado donde liste los diferentes tipos de gráficos que se puede mostrar según el total de boletas por cliente, tal como se muestra en la siguiente imagen:



- El gráfico deberá mostrarse en el lado derecho de la tabla, tal como se muestra en la imagen inicial de la aplicación.
- El gráfico de barras debe mostrarse como la siguiente imagen:

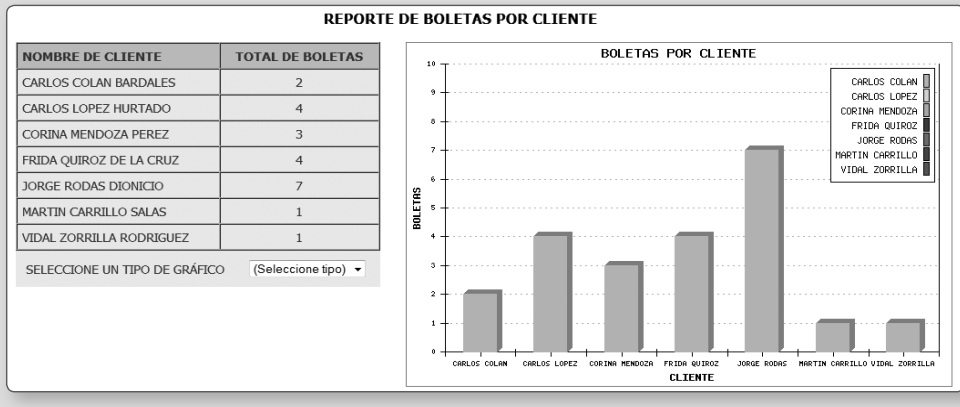


Imagen: listado.php

- El gráfico pastel debe mostrarse como la siguiente imagen:

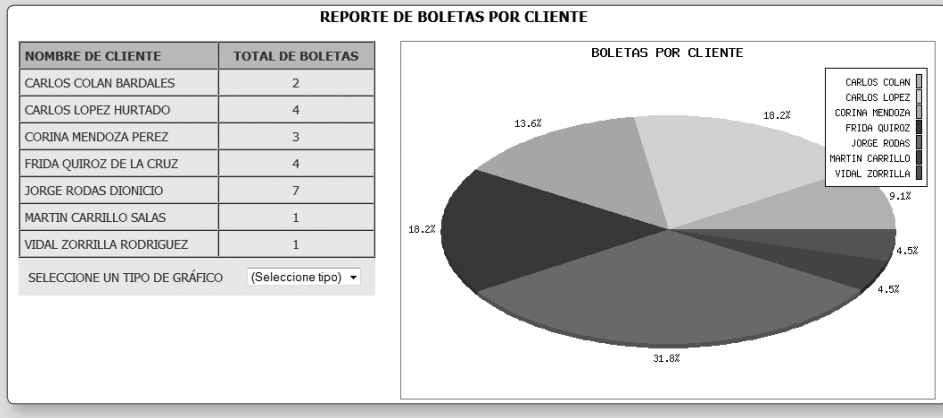


Imagen: listado.php

- El gráfico líneas debe mostrarse como la siguiente imagen:

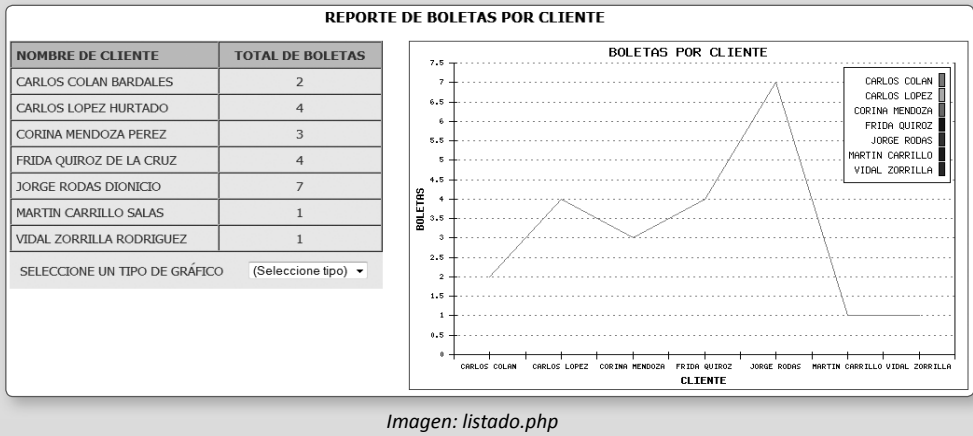


Imagen: listado.php

- El gráfico horizontal debe mostrarse como la siguiente imagen:

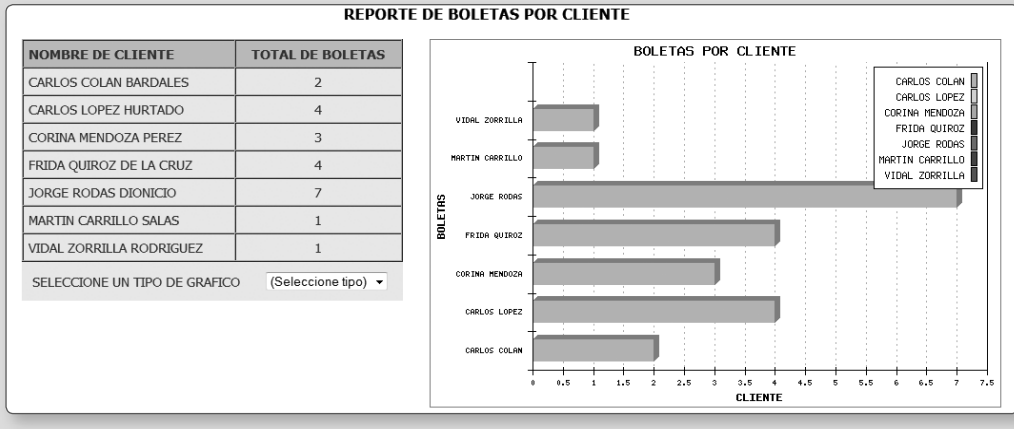


Imagen: listado.php

- El gráfico puntos debe mostrarse como la siguiente imagen:

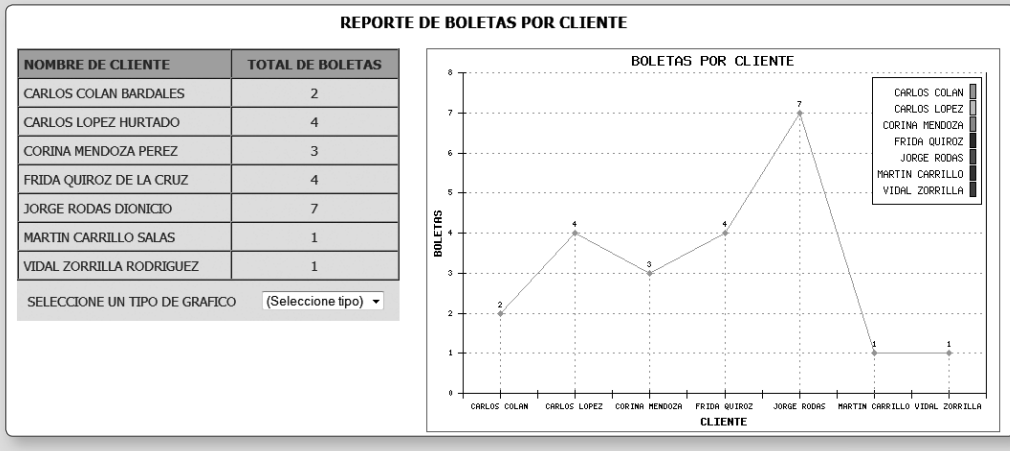


Imagen: listado.php

El objetivo de cada archivo que contiene el proyecto es:

- **conecta.php:** Contiene las sentencias que permiten conectarse al servidor MySQL y seleccionar la base de datos Ventas2015.
- **encabezado.php:** Contiene el encabezado del documento principal.
- **listado.php:** Contiene el archivo web que interactúa con el usuario, desde aquí se reportan los datos del cliente y el total de boletas asociadas a dicho cliente; además, mostrará el gráfico según el tipo seleccionado por el usuario.
- **phplot.php:** Contiene todas las funciones que permiten implementar gráficos.
- **barras.php:** Contiene las sentencias necesarias para crear el gráfico de barras y mostrarlo en una etiqueta IMG, este archivo será mostrado en **listado.php**.
- **pastel.php:** Contiene las sentencias necesarias para crear el gráfico de pastel y mostrarlo en una etiqueta IMG, este archivo será mostrado en **listado.php**.
- **lineas.php:** Contiene las sentencias necesarias para crear el gráfico de líneas y mostrarlo en una etiqueta IMG, este archivo será mostrado en **listado.php**.

- **horizontal.php**: Contiene las sentencias necesarias para crear el gráfico de horizontal y mostrarlo en una etiqueta IMG, este archivo será mostrado en **listado.php**.
- **puntos.php**: Contiene las sentencias necesarias para crear el gráfico de puntos y mostrarlo en una etiqueta IMG, este archivo será mostrado en **listado.php**.
- **miEstilo.css**: Contiene los estilos usados en el archivo **listado.php**.

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **encabezado.php**

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <img src="banner.jpg" width="920" height="230"/>
        <h3> REPORTE DE BOLETAS POR CLIENTE </h3>
    </body>
</html>
```

Archivo: **listado.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">
  </head>
  <body>
    <header>
       <?php
           include('encabezado.php');
           include('conecta.php');
        ?>
    </header>
    <section>

      <article id="columna">
        <form method="POST">
         <table border="1" width="400" cellspacing="0" cellpadding="5">
          <tr>
           <th>NOMBRE DE CLIENTE</th>
           <th id="centrado">TOTAL DE BOLETAS</th>
          </tr>
          <?php
            $rs=mysql_query("SELECT  CONCAT(C.NOMBRES,'  ',C.PATERNO,'  ',C.MATERNO)  AS
            CLIENTE,
                                COUNT(*) AS TOTAL
                                FROM BOLETA B
                                INNER JOIN CLIENTE C ON B.ID_CLIENTE=C.ID_CLIENTE
                              GROUP BY CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO);",$cn);
```

```
            while($boletas = mysql_fetch_array($rs)){
          ?>
          <tr>
            <td> <?php echo utf8_decode($boletas[0]); ?> </td>
            <td id="centrado"><?php echo $boletas[1]; ?> </td>
          </tr>
          <?php } ?>
        </table>
        <table border="0" width="400" cellspacing="1" cellpadding="10">
          <tr>
            <td>SELECCIONE UN TIPO DE GRAFICO</td>
            <td><select name="selTipo" onchange="this.form.submit()">
                      <option value="0">(Seleccione tipo)</option>
                      <option value="1">BARRAS</option>
                      <option value="2">PASTEL</option>
                      <option value="3">LINEAS</option>
                      <option value="4">HORIZONTAL</option>
                      <option value="5">PUNTOS</option>
                </select>
             </td>
           </tr>
         </table>
       </form>
      </article>
      <article id="columna">
        <?php
           if(isset($_POST['selTipo'])) {
              if($_POST['selTipo']==1) {
        ?>
                   <img src="barras.php" >
        <?php   } elseif($_POST['selTipo']==2) { ?>
                   <img src="pastel.php" >
        <?php } elseif($_POST['selTipo']==3) { ?>
                   <img src="lineas.php" >
        <?php }elseif($_POST['selTipo']==4) { ?>
                   <img src="horizontal.php" >
        <?php }elseif($_POST['selTipo']==5) { ?>
                   <img src="puntos.php" >
        <?php }
           }
        ?>
       </article>
     </section>
  </body>
</html>
```

Comentarios

```
<?php
  $rs=mysql_query("SELECT CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
                   COUNT(*) AS TOTAL
                   FROM BOLETA B
                   INNER JOIN CLIENTE C ON B.ID_CLIENTE=C.ID_CLIENTE
                   GROUP BY CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO);",$cn);
  while($boletas = mysql_fetch_array($rs)){
?>
```

La variable **$rs** permite ejecutar la consulta en la cual se une los nombres, paterno y materno del cliente; se debe tener en cuenta que para unir las columnas se usa la función **CONCAT** de MySQL y, además, el total de boletas asociadas al cliente, para esto se debe agrupar por la unión de sus nombres, paterno y materno.

Se usa la estructura repetitiva **while** para recorrer por todos los registros obtenidos en la consulta mediante la variable **$rs** y el uso de la función **mysql_fetch_array**, esta información permitirá imprimir la información en la tabla correspondiente.

```
<tr>
   <td> <?php echo utf8_decode($boletas[0]); ?> </td>
   <td id="centrado"><?php echo $boletas[1]; ?> </td>
</tr>
<?php } ?>
```

A partir de la variable **$boletas** se obtiene toda la información que se necesita mostrar, como el nombre completo del cliente (**boletas[0]**) y el total de boletas por cliente (**boletas[1]**), estos serán impresos en las celdas correspondientes.

```
<td><select name="selTipo" onchange="this.form.submit()">
          <option value="0">(Seleccione tipo)</option>
          <option value="1">BARRAS</option>
          <option value="2">PASTEL</option>
          <option value="3">LINEAS</option>
          <option value="4">HORIZONTAL</option>
          <option value="5">PUNTOS</option>
    </select>
</td>
```

Se implementa la etiqueta SELECT para los diferentes tipos de gráficos que presentará la aplicación, de forma que al seleccionar un tipo mostrará la imagen en el lado derecho, para esto se activa la propiedad onchange="this.form.submit()".

```
<?php
    if(isset($_POST['selTipo'])) {
      if($_POST['selTipo']==1) {
?>
        <img src="barras.php" >
<?php } elseif($_POST['selTipo']==2) { ?>
        <img src="pastel.php" >
<?php } elseif($_POST['selTipo']==3) { ?>
        <img src="lineas.php" >
<?php }elseif($_POST['selTipo']==4) { ?>
        <img src="horizontal.php" >
<?php }elseif($_POST['selTipo']==5) { ?>
        <img src="puntos.php" >
<?php }
    }
?>
```

Al iniciar la aplicación no se debe mostrar imagen alguna, ya que el usuario primero deberá seleccionar un tipo de imagen; para esto usamos la función **if(isset($_POST['selTipo']))**.

Para determinar el tipo de gráfico que se desea mostrar, se captura el tipo 1 y se muestra la imagen mediante la etiqueta IMG. Tomar en cuenta que **barras.php** genera una imagen, es así que se puede imprimir la imagen en la propiedad SCR de la etiqueta IMG. De la misma manera, mostraremos los demás gráficos asociados a un tipo seleccionado por el usuario.

Archivo: **barras.php**

```
<?php
    include('conecta.php');
    include('phplot.php');

    $data=array();
    $gra=new PHPlot();

    //Implementación de la consulta
    $sql="SELECT CONCAT(C.NOMBRES,' ',C.PATERNO),
            COUNT(*)
            FROM BOLETA B
            INNER JOIN CLIENTE C ON B.ID_CLIENTE=C.ID_CLIENTE
            GROUP BY CONCAT(C.NOMBRES,' ',C.PATERNO)";
    $rs=mysql_query($sql, $cn);

    //Obteniendo los valores desde la consulta
    while ($row = mysql_fetch_row($rs)) {
        $data[]=array(utf8_decode($row[0]),$row[1]);
        $cols[]=utf8_decode($row[0]);
    }
    //Enviando los valores al gráfico tipo vertical
    $gra->SetImageBorderType('plain');
    $gra->SetDataValues($data);
    $gra->SetLineStyles("solid");
    $gra->SetPlotType("bars");

    //Especificando los títulos de la imagen
    $gra->SetTitle("BOLETAS POR CLIENTE");
    $gra->SetXTitle("CLIENTE");
    $gra->SetYTitle("BOLETAS");
    //Especificando la leyenda
    $gra->SetLegend($cols);

    //Especificando la proyección del área del gráfico
    $gra->SetPlotÁreaWorld(NULL,0, NULL, 10);
    $gra->DrawGraph();
?>
```

Archivo: **pastel.php**

```
<?php
    include('conecta.php');
    include('phplot.php');

    $data=array();
    $gra=new PHPlot();

    //Implementación de la consulta
    $sql="SELECT CONCAT(C.NOMBRES,' ',C.PATERNO),
            COUNT(*)
            FROM BOLETA B
            INNER JOIN CLIENTE C ON B.ID_CLIENTE=C.ID_CLIENTE
            GROUP BY CONCAT(C.NOMBRES,' ',C.PATERNO)";
    $rs=mysql_query($sql, $cn);

    //Obteniendo los valores desde la consulta
    $data[]='0';
    while ($row = mysql_fetch_row($rs)) {
        $data[]=$row[1];
        $cols[]=utf8_decode($row[0]);
```

```
    }

    //Enviando los valores al gráfico de tipo Pastel
    $gra->SetImageBorderType('plain');
    $gra->SetDataValues(array($data));
    $gra->SetLineStyles("solid");
    $gra->SetPlotType("pie");

    //Especificando los títulos de la imagen
    $gra->SetTitle("BOLETAS POR CLIENTE");
    $gra->SetXTitle("CLIENTE");
    $gra->SetYTitle("BOLETAS");

    //Especificando la leyenda
    $gra->SetLegend($cols);

    //Especificando la proyección del área del gráfico
    $gra->SetPlotÁreaWorld(NULL,0, NULL, 5);
    $gra->DrawGraph();
?>
```

Archivo: **lineas.php**

```
<?php
    include('conecta.php');
    include('phplot.php');

    $data=array();
    $gra=new PHPlot();

    //Implementación de la consulta
    $sql="SELECT CONCAT(C.NOMBRES,' ',C.PATERNO),
            COUNT(*)
            FROM BOLETA B
            INNER JOIN CLIENTE C ON B.ID_CLIENTE=C.ID_CLIENTE
            GROUP BY CONCAT(C.NOMBRES,' ',C.PATERNO)";
    $rs=mysql_query($sql, $cn);

    //Obteniendo los valores desde la consulta
    while ($row = mysql_fetch_row($rs)) {
        $data[]=array(utf8_decode($row[0]),$row[1]);
        $cols[]=utf8_decode($row[0]);
    }

    //Enviando los valores al gráfico de tipo línea
    $gra->SetImageBorderType('plain');
    $gra->SetDataValues($data);
    $gra->SetLineStyles("solid");
    $gra->SetPlotType("lines");

    //Especificando los títulos de la imagen
    $gra->SetTitle("BOLETAS POR CLIENTE");
    $gra->SetXTitle("CLIENTE");
    $gra->SetYTitle("BOLETAS");

    //Especificando la leyenda
    $gra->SetLegend($cols);

    //Especificando la proyección del área
    $gra->SetPlotÁreaWorld(NULL, 0, NULL, NULL);
    $gra->DrawGraph();
?>
```

Archivo: **horizontal.php**

```
<?php
    include('conecta.php');
    include('phplot.php');

    $data=array();
    $gra=new PHPlot();

    //Implementación de la consulta
    $sql="SELECT CONCAT(C.NOMBRES,' ',C.PATERNO),
            COUNT(*)
            FROM BOLETA B
            INNER JOIN CLIENTE C ON B.ID_CLIENTE=C.ID_CLIENTE
            GROUP BY CONCAT(C.NOMBRES,' ',C.PATERNO)";
    $rs=mysql_query($sql, $cn);

    //Obteniendo los valores desde la consulta
    while ($row = mysql_fetch_row($rs)) {
        $data[]=array(utf8_decode($row[0]),$row[1]);
        $cols[]=utf8_decode($row[0]);
    }
    //Enviando los valores al gráfico de tipo barra horizontal
    $gra->SetImageBorderType('plain');
    $gra->SetDataValues($data);
    $gra->SetLineStyles("solid");
    $gra->SetDataType('text-data-yx');
    $gra->SetPlotType('bars');

    //Especificando los títulos de la imagen
    $gra->SetTitle("BOLETAS POR CLIENTE");
    $gra->SetXTitle("CLIENTE");
    $gra->SetYTitle("BOLETAS");

    //Especificando la leyenda
    $gra->SetLegend($cols);

    //Especificando la proyección del área
    $gra->SetPlotÁreaWorld(NULL,0, NULL, 8);
    $gra->DrawGraph();
?>
```

Archivo: **puntos.php**

```
<?php
    include('conecta.php');
    include('phplot.php');

    $data=array();
    $gra=new PHPlot();

    //Implementación de la consulta
    $sql="SELECT CONCAT(C.NOMBRES,' ',C.PATERNO),
            COUNT(*)
            FROM BOLETA B
            INNER JOIN CLIENTE C ON B.ID_CLIENTE=C.ID_CLIENTE
            GROUP BY CONCAT(C.NOMBRES,' ',C.PATERNO)";
    $rs=mysql_query($sql, $cn);

    //Obteniendo los valores desde la consulta
    while ($row = mysql_fetch_row($rs)) {
        $data[]=array(utf8_decode($row[0]),$row[1]);
        $cols[]=utf8_decode($row[0]);
```

```
    }

    //Enviando los valores al gráfico de tipo puntos
    $gra->SetImageBorderType('plain');
    $gra->SetDataValues($data);
    $gra->SetLineStyles("solid");
    $gra->SetPlotType('linepoints');

    //Mostrando los valores en cada punto
    $gra->SetYDataLabelPos('plotin');

    //Mostrando grilla de la imagen
    $gra->SetDrawXDataLabelLines(True);
    //Especificando los títulos de la imagen
    $gra->SetTitle("BOLETAS POR CLIENTE");
    $gra->SetXTitle("CLIENTE");
    $gra->SetYTitle("BOLETAS");
    //Especificando la leyenda
    $gra->SetLegend($cols);
    //Especificando la proyección del área
    $gra->SetPlotÁreaWorld(NULL,0, NULL, 8);
    $gra->DrawGraph();
?>
```

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
#centrado{
    text-align: center;
}
#izquierda{
    text-align: left;
}
table {
    margin: auto;
    background-color: antiquewhite;
}
img{
    margin: auto;
    display: block;
}
table th {
background-color: #b9c9fe;
}
#columna
{
    position:relative;
    float:left;
    left:200px;
    width: 30%;
    top:0px;
    text-align:justify;
    color:#039;
    padding: 0px 5px 0px 10px;
    font-family:Tahoma, Geneva, sans-serif;
}
```

Caso desarrollado 4 Gráfico con Google Charts-Listado de boletas por cliente

Implementar una aplicación web con PHP y MySQL que permita reportar el número total de boletas registradas por cliente, mostrando un menú de opciones para graficar en forma de pastel y líneas los valores usando Google Charts, tal como se muestra en la siguiente imagen:

Listado | Gráfico Pastel | Gráfico Líneas

LISTADO DE BOLETAS POR CLIENTE

NOMBRE DE CLIENTE	TOTAL DE BOLETAS
CARLOS COLAN BARDALES	2
CARLOS LOPEZ HURTADO	4
CORINA MENDOZA PEREZ	3
FRIDA QUIROZ DE LA CRUZ	4
JORGE RODAS DIONICIO	7
MARTIN CARRILLO SALAS	1
VIDAL ZORRILLA RODRIGUEZ	1

Todos los derechos reservados a @manueltorres-2015

Lima-Perú

Imagen: index.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **GoogleCharts**.
- Al iniciar la aplicación deberá mostrar la página **index.php**.
- Descargar la librería necesaria para la implementación de los gráficos Google Charts desde la página https://code.google.com/p/googchart/downloads/list. Una vez descargada la librería copie el archivo **GoogChart.class.php** al proyecto.

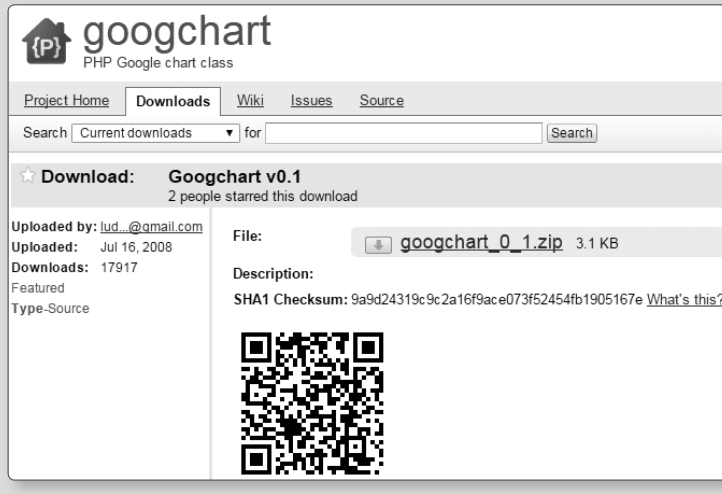


Imagen: https://code.google.com/p/googchart/downloads/detail?name=googchart_0_1.zip&can=2&q

- Para la generación del listado y sus imágenes se necesita:

  - Un listado de los clientes con su respectivo número de boletas asociadas, tal como se muestra en la siguiente imagen:

Imagen: listado.php

  - El gráfico tipo pastel muestra a los clientes y la cantidad de boletas asociadas, tal como se muestra en la siguiente imagen:

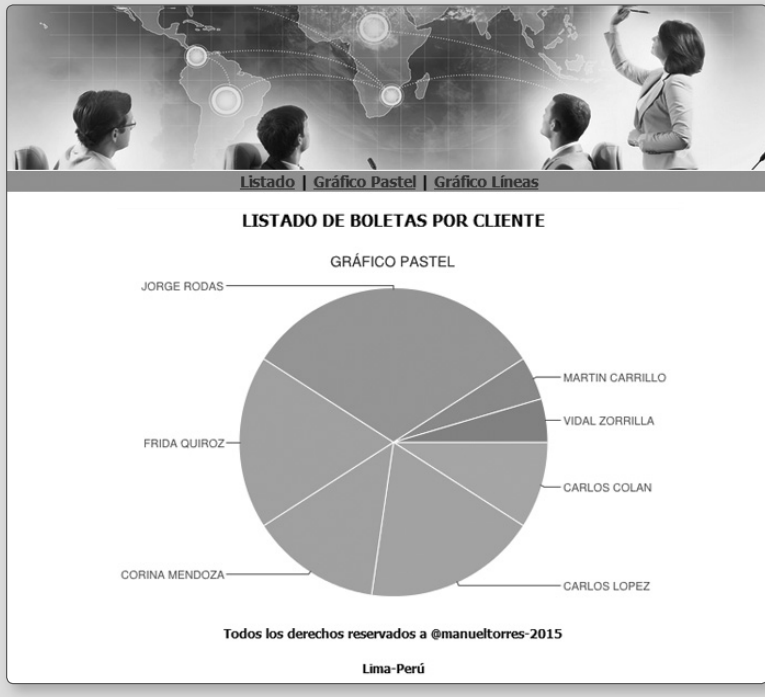


Imagen: pastel.php

- El gráfico tipo línea muestra a los clientes y la cantidad de boletas asociadas, tal como se muestra en la siguiente imagen:

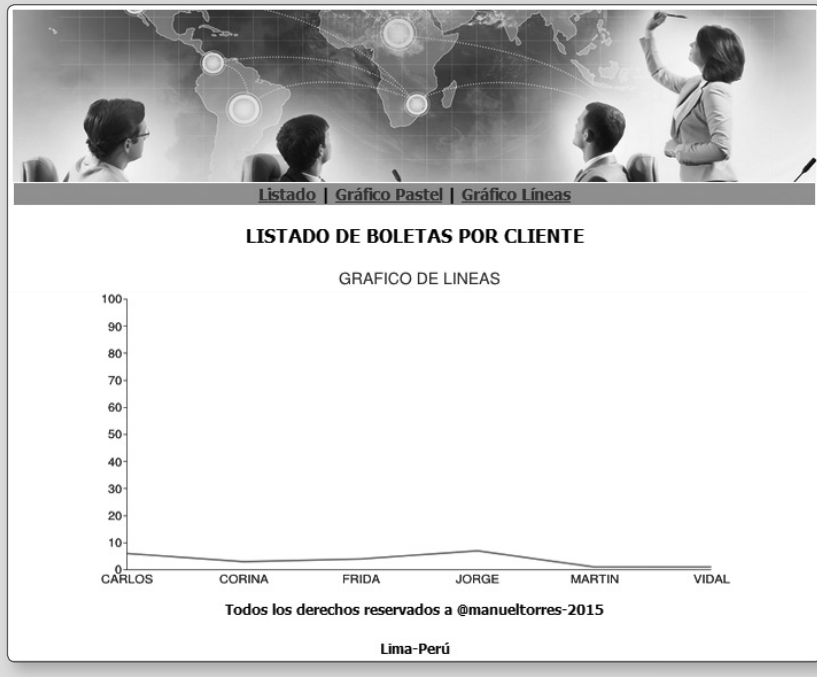
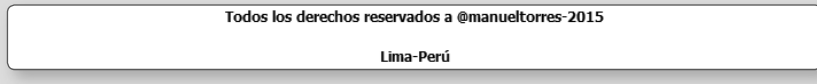


Imagen: líneas.php

El objetivo de cada archivo que contiene el proyecto es:

- **conecta.php:** Contiene las sentencias que permiten conectarse al servidor MySQL y selecciona la base de datos Ventas2015.
- **GoogChart.class.php:** Contiene las funciones necesarias para la implementación de los gráficos con Google Charts.
- **index.php:** Contiene el archivo web que interactúa con el usuario.
- **encabezado.php:** Contiene el encabezado de todos los documentos.

- **pie.php**: contiene el encabezado del documento principal.



- **listado.php**: Contiene el archivo web que lista los clientes con el número total de las boletas.
- **pastel.php**: Muestra la imagen de tipo pastel con los datos de los clientes y el total de boletas.

- **lineas.php**: Muestra la imagen de tipo líneas con los datos de los clientes y el total de boletas.
- **miEstilo.css**: Contiene los estilos usados en todos los archivos del proyecto.

Archivo: **conecta.php**

```
<?php
    $cn = mysql_connect('localhost','root','');
    mysql_select_db('Ventas2015',$cn);
?>
```

Archivo: **index.php**

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
        <link href="miEstilo.css" rel="stylesheet">
    </head>
    <body>
        <header>
            <?php include('encabezado.php'); ?>
        </header>
        <section>
            <?php header('location:listado.php'); ?>
        </section>
        <footer>
            <?php include('pie.php'); ?>
        </footer>
    </body>
</html>
```

Archivo: **listado.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">
  </head>
  <body>
    <header>
       <?php
         include('encabezado.php');
         include('conecta.php');
       ?>
       <h3>LISTADO DE BOLETAS POR CLIENTE</h3>
    </header>
    <section>
      <table border="1" width="700" cellspacing="0" cellpadding="5">
        <tr>
           <th>NOMBRE DE CLIENTE</th>
           <th id="centrado">TOTAL DE BOLETAS</th>
        </tr>
        <?php
           //Implementando la consulta que permita mostrar el total de boletas
```

```
            $rs=mysql_query("SELECT
                    CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
                    COUNT(*) AS TOTAL
                    FROM BOLETA B
                    JOIN CLIENTE C ON B.ID_CLIENTE=C.ID_CLIENTE
                GROUP BY CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO);",$cn);

            //Recorrer por los registros y mostrarlos en la tabla
            while($boletas = mysql_fetch_array($rs)){
          ?>
          <tr>
            <td><?php echo utf8_decode($boletas[0]); ?></td>
            <td id="centrado"><?php echo $boletas[1]; ?></td>
          </tr>
          <?php } ?>
        </table>
      </section>
      <footer>
        <?php include('pie.php'); ?>
      </footer>
    </body>
</html>
```

Archivo: **pastel.php**

```
<?php
  include('GoogChart.class.php');
  include('conecta.php');

  $chart = new GoogChart();

  //Implementación de la consulta
  $sql="SELECT CONCAT(C.NOMBRES,' ',C.PATERNO),
            COUNT(*)
            FROM BOLETA B
            INNER JOIN CLIENTE C ON B.ID_CLIENTE=C.ID_CLIENTE
            GROUP BY CONCAT(C.NOMBRES,' ',C.PATERNO);";
  $rs=mysql_query($sql, $cn);

  //Creando un arreglo a partir de la consulta
  while ($row = mysql_fetch_row($rs)) {
$data[$row[0]]=$row[1];
  }

  //Definición de los colores de la imagen
  $color = array('#99C754','#54C7C5','#999999');

  //Parámetros de implementación del gráfico
  $chart->setChartAttrs( array(
                    'type' => 'pie',
                    'title' => 'GRÁFICO PASTEL',
                    'data' => $data,
                    'size' => array(600, 350),
                    'color' => $color
       ));
?>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
```

```
        <title></title>
        <link href="miEstilo.css" rel="stylesheet">
    </head>
    <body>
        <header>
            <?php include('encabezado.php'); ?>
            <h3>LISTADO DE BOLETAS POR CLIENTE</h3>
        </header>
        <section>
            <p id="centrado"><?php echo $chart; ?></p>
        </section>
        <footer>
            <?php include('pie.php'); ?>
        </footer>
    </body>
</html>
```

Archivo: **lineas.php**

```
<?php
    include('GoogChart.class.php');
    include('conecta.php');
    $chart = new GoogChart();
    $color = array('#99C754',
                   '#54C7C5',
                   '#999999');

    //Implementación de la consulta
    $sql="SELECT C.NOMBRES,
            COUNT(*)
            FROM BOLETA B
            INNER JOIN CLIENTE C ON B.ID_CLIENTE=C.ID_CLIENTE
            GROUP BY C.NOMBRES";
    $rs=mysql_query($sql, $cn);

    //Creando un arreglo a partir de la consulta
    while ($row = mysql_fetch_row($rs)) {
        $data[$row[0]]=$row[1];
    }

    $chart->setChartAttrs( array(
                            'type' => 'line',
                            'title' => 'GRAFICO DE LINEAS',
                            'data' => $data,
                            'size' => array( 550, 300 ),
                            'color' => $color,
                            'labelsXY' => true,
            ));
?>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
        <link href="miEstilo.css" rel="stylesheet">
    </head>
    <body>
        <header>
            <?php include('encabezado.php'); ?>
```

```
            <h3>LISTADO DE BOLETAS POR CLIENTE</h3>
        </header>
        <section>
            <p id="centrado"><?php echo $chart; ?></p>
        </section>
        <footer>
            <?php include('pie.php'); ?>
        </footer>
    </body>
</html>
```

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
#centrado{
    text-align: center;
}
table th {
    background-color: coral;
    text-align: left;
}
table {
    margin: auto;
}
img{
    margin: auto;
    display: block;
}
```

Programación orientada a objetos

6.1 Introducción

La programación orientada a objetos surge como un intento de dominar la complejidad que tiene la construcción de un *software* de aplicaciones. Es decir, ahora se desarrollarán aplicaciones grandes, medianas o pequeñas, de igual manera, duraderas en el tiempo y factibles a las modificaciones.

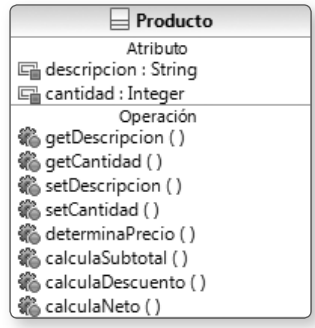
Tradicionalmente, la forma de enfrentar la complejidad del *software* era usando la programación estructurada, la cual hacía que una aplicación compleja se descomponga en porciones pequeñas de códigos simples y fáciles de codificar. Entonces, se puede decir que la programación orientada a objetos es otra forma de descomponer el código por medio de la descomposición, pero de objetos.

6.2 Clases en PHP

A. Definición

Una clase es un elemento básico de la programación orientada a objetos, representa una plantilla que permite definir las características similares que pueden presentar los objetos.

Entonces, se puede decir que una clase es una declaración de objetos; en otras palabras, la definición de un objeto es una clase. Cuando se programa sobre un objeto y se le define atributos y operaciones, en realidad lo que se está haciendo es programar en una clase.



B. Partes de una clase

En el lenguaje de modelamiento unificado, la clase es representada por tres compartimientos, como se visualiza en la siguiente tabla:

NOMBRE DE LA CLASE
Lista de atributos
Lista de operaciones

En PHP se visualiza con el siguiente script:

```
class nombre_de_la_clase
{
    Lista de atributos
    Lista de operaciones
}
```

Nivel de la clase

C. Atributos de clase

Un atributo de clase es la característica que presenta un objeto. Su formato es:

```
Visibilidad $nombreAtributo;
```

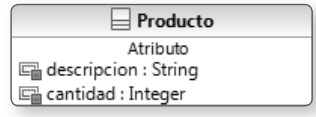
- Visibilidad: Puede ser private, public o sin definición, normalmente los atributos de clases son implementados con visibilidad privada, ya que por medio de los métodos GET y SET se puede acceder a ellas.
- $nombreAtributo: Es el nombre que se le asignará al atributo de clase, se debe tener especial cuidado con la sensibilidad de mayúsculas y minúsculas.

Ejemplos de declaración de atributos:

Suponiendo que se tiene la clase Producto y se necesite declarar el atributo descripción, cuántas posibilidades se tiene para declararla y cuál es la consecuencia de ello:

Declaración de un atributo	Explicación
$descripcion="";	Al declarar el atributo $descripcion sin visibilidad, solo podrá ser accedida desde la misma clase y no desde el exterior.
public $descripcion="";	Al declarar el atributo $descripcion como públic, todas las clases del mismo proyecto podrán hacer referencia al valor contenido en dicho atributo.
private $descripcion="";	Al declarar el atributo $descripcion como private, solo podrá ser accedida en la clase que se implementó. Cabe mencionar que la implementación del atributo como privado es lo más recomendado.

Veamos el script que declara los atributos de la clase Producto como privados:



En PHP se tendría el siguiente script:

```
<?php
  class Producto{
        private $descripcion='';
        private $cantidad=0;
  }
?>
```

Se inicializa los atributos con valores relativos a su declaración, la descripción del producto se inicializa con comillas simples o dobles por ser de tipo cadena y la cantidad es declarada como número entero, por lo tanto, se inicializa con el número 0.

D. Operaciones de clase

También son llamadas métodos de clase, estos tienen el mismo propósito que las funciones de una aplicación.

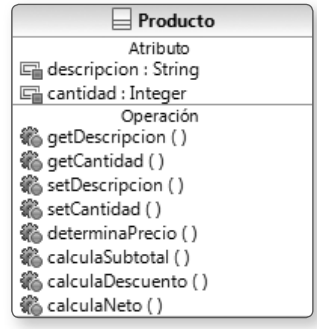
```
Visibilidad nombreMetodo();
Visibilidad nombreMetodo($parámetros);
```

- visibilidad: Puede ser private, public o sin definición.
- nombreMetodo(): Es el nombre que se le asignará al método de clase, se debe tener en cuenta que dependiendo del trabajo que realice el método se deberán implementar los parámetros.

Veamos un ejemplo de implementación del método getDescripcion de la clase Producto:

DECLARACIÓN DE UN MÉTODO	EXPLICACIÓN
`function getDescripcion(){ }`	Se declara el método getDescripcion sin visibilidad; por lo tanto, solo será accesible dentro de la clase. Tenga en cuenta que todo método en PHP debe iniciar con la palabra reservada function.
`public function getDescripcion(){ }`	Al declarar el método getDescripcion como público, podrá ser accesible desde todas las clases de la misma aplicación.

Si se necesita implementar los métodos de la clase Producto en un sistema de ventas, se tendría:



En PHP se tendría el siguiente script de la clase Producto:

```
<?php
    class producto{
        private $descripcion='';
        private $cantidad=0;

        public function setDescripcion($descripcion){}
        public function getdescripcion(){}
        public function setCantidad($cantidad){}
        public function getCantidad(){}
```

```
        public function determinaPrecio(){}
        public function calculaSubtotal(){}
        public function calculaDescuento(){}
        public function calculaNeto(){}
    }
?>
```

6.3 Encapsulamiento de datos

A. Definición

Se le llama encapsulamiento de datos a la forma en que los atributos de la clase sean visibles para las demás clases, la forma de implementar el encapsulamiento es crear métodos GET y SET de cada uno de los atributos privados de la clase.

B. Creación de método SET

Los métodos SET se encargan de asignar valores a los atributos privados de la clase, estos valores serán enviados por medio de parámetros desde una clase exterior.

```
public function setAtributo($valor){
$this->atributo= $valor;
}
```

- setAtributo: Es el nombre del método SET para un determinado atributo.
- $valor: Aquí se especifica el valor proveniente del exterior que registrará su dato en el atributo privado de la clase, se debe tener en cuenta que cada atributo genera un método SET por cada uno; por lo tanto, no será dable colocar dos valores en la definición de los métodos SET.
- $this: Es el operador que hace referencia a la clase actual, desde aquí se puede tener acceso a los atributos o métodos implementados en su misma clase.

C. Creación de método GET

Los métodos GET se encargan de enviar la información almacenada en los atributos privados de la clase, se debe tener en cuenta que estos valores fueron obtenidos gracias a los métodos SET.

```
public function getAtributo(){
return $this->atributo;
}
```

- getAtributo(): Es el nombre del método GET para un determinado atributo.
- return: Es la cláusula que permite especificar cuál es la información devuelta por el método GET, normalmente se especifican los atributos privados de la clase.
- atributo: Es el nombre del atributo privado de la clase que necesita ser evidente frente a las demás clases.

6.4 Objetos en PHP

A. Definición

Los objetos son copias o ejemplares de una clase. El hecho de crear un objeto permite tener acceso a los atributos y métodos definidos en la clase. Para nuestras aplicaciones en PHP, los objetos se crearán en el formulario de registro de datos.

B. Creación de objetos

```
$objeto = new clase();
```

- $objeto: Es el nombre que se le asigna al objeto de clase, este será el enlace con los elementos que compone la clase.
- clase(): Aquí se debe especificar la clase de donde proviene el objeto.
- new: Es el operador usado por PHP para crear un objeto.

C. Referenciar a los métodos de una clase

```
$objeto -> método();
$objeto -> método($parámetros);
```

- $objeto: El objeto es la referencia a la clase, el guion más el signo mayor es obligatorio al momento de invocar algún método público de la clase.
- método(): Es el nombre de los métodos implementados en la clase.

6.5 Método constructor en PHP

A. Definición

En PHP un método constructor permite inicializar los atributos dentro de un objeto de clase; es decir, este método es el primero en ejecutarse al crear un objeto.

Asimismo, presenta ciertas características que lo diferencian de los métodos, por ejemplo, que la invocación al método constructor se realiza automáticamente, no permite retomar valores, puede o no recibir parámetros y principalmente su implementación es opcional.

B. Implementacion de un método constructor

```
public function__construct($parametros) {
  $this->atributo = $parametro;
}
```

- __construct: Es el nombre de implementación del método constructor.
- $parametros: Aquí se especifican los parámetros que necesita inicializar el método constructor.
- $this->atributo=$parametro: Es la expresión que permite asignar el valor que proviene del parámetro al atributo privado de la clase, recuerde que dicho atributo es privado y no es accesible directamente por el usuario.

Por ejemplo, veamos cómo implementar un método constructor para la clase libro el cual cuenta con idLibro, descripción, autor, editorial y fechaRegistro

```
public function__construct($idLibro,$descripcion,$autor,$editorial,$fechaRegistro) {
        $this->idLibro=$idLibro;
        $this->descripcion=$descripcion;
        $this->autor=$autor;
        $this->editorial=$editorial;
        $this->fechaRegistro=$fechaRegistro;
}
```

6.6 Casos desarrollados

Caso desarrollado 1 **Clase/objeto-Venta de productos**

Implementar una aplicación web con PHP que permita realizar la venta de productos usando la programación orientada a objetos, tal como se muestra en la siguiente imagen:

Imagen: index.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **ClaseObjeto**.
- Al iniciar la aplicación deberá mostrar la página **index.php**.
- Al iniciar la aplicación tener en cuenta los siguientes aspectos:

  - Mostrar los siguientes productos en el control cuadro combinado:

    Videograbadora
    Cámara
    Televisor
    Radiograbadora

- Al presionar el botón **PROCESAR** se debe mantener tanto el producto seleccionado en el cuadro combinado como la cantidad comprada y, finalmente, mostrar los resultados como subtotal (cantidad por precio), descuento (10 % del subtotal) y el neto a pagar (diferencia entre el subtotal y el descuento):

El objetivo de cada archivo que contiene el proyecto es:

- **producto.Php:** Es considerado la clase Producto, el cual contiene atributos, métodos GET y SET, y los métodos propios de la aplicación como determinar el precio de los productos, cálculo del subtotal, cálculo del descuento y cálculo del neto.
- **index.php:** Contiene la página web que interactúa con el usuario para gestionar una venta de productos.
- **miEstilo.css:** Contiene los estilos usados en el archivo **index.php**.

Archivo: **producto. Php**

```
<?php
    //Implementación de la clase producto
    class producto{
        //Atributos de clase
        private $descripcion='';
        private $cantidad=0;

        //Encapsulado de atributos
        public function setDescripcion($descripcion){
            $this->descripcion = $descripcion;
        }

        public function getDescripcion(){
            return $this->descripcion;
        }
        public function setCantidad($cantidad){
            $this->cantidad = $cantidad;
        }

        public function getCantidad(){
            return $this->cantidad;
        }

        //Métodos propios
        public function determinaPrecio(){
            if ($this->descripcion== 'Videograbadora')
                return 1500;
                elseif ($this->descripcion == 'Camara')
                    return 900;
                elseif ($this->descripcion == 'Televisor')
                    return 2500;
                elseif ($this->descripcion == 'Radiograbadora')
                    return 500;
                else
                    return 0;
        }

        public function calculaSubtotal(){
            return $this->determinaPrecio() * $this->cantidad;
        }

        public function calculaDescuento(){
            return $this->calculaSubtotal()*0.1;
        }

        public function calculaNeto(){
            return $this->calculaSubtotal()-$this->calculaDescuento();
        }
    }
?>
```

Comentarios

```
private $descripcion='';
private $cantidad=0;
```

Se declara los atributos de clase para la descripción del producto ($descripcion) y la cantidad comprada por el cliente ($cantidad); no se declara el precio del producto ya que existe un método que asignará los precios según el producto seleccionado. Ambos atributos han sido inicializados según la naturaleza de sus datos. Recuerde que los atributos de clase son declarados como privados.

```
public function setDescripcion($descripcion){
    $this->descripcion = $descripcion;
}

public function getDescripcion(){
    return $this->descripcion;
}

public function setCantidad($cantidad){
    $this->cantidad = $cantidad;
}

public function getCantidad(){
    return $this->cantidad;
}
```

Implementar los métodos GET y SET para cada atributo privado; el nombre de los métodos inicializarán con las palabras SET o GET, respectivamente, seguido del nombre del atributo. Recuerde que los métodos SET tienen la misión de asignar un valor al atributo privado, por tal motivo, siempre tendrá que recibir un parámetro y al momento de asignar dicho valor debe usar el operador **$this** para hacer referencia a los atributos de la misma clase.

En caso de los métodos GET, solo devuelven el valor asignado a los atributos privados de la clase, haciendo que dichos valores sean evidentes para las clases que se asocian.

Finalmente, todos los métodos GET y SET implementados en una clase, deben ser de visibilidad pública, para que todas las clases puedan tener acceso a dichos métodos.

```
public function determinaPrecio(){
     if ($this->descripcion== 'Videograbadora')
        return 1500;
        elseif ($this->descripcion == 'Camara')
        return 900;
        elseif ($this->descripcion == 'Televisor')
        return 2500;
        elseif ($this->descripcion == 'Radiograbadora')
        return 500;
        else
        return 0;
}
```

La función **determinaPrecio** tiene por misión asignar un precio al producto seleccionado por el usuario, este producto seleccionado ha sido registrado en el atributo **descripción** gracias al método **SET**. Para poder comparar cada tipo de producto se debe evaluar la descripción usando la estructura IF; una vez comparado, enviar el precio por producto mediante la cláusula RETURN. Recuerde que para referenciar al atributo, se debe usar el operador **$this** seguido del símbolo -> y, finalmente, el nombre del atributo.

```
public function calculaSubtotal(){
    return $this->determinaPrecio() * $this->cantidad;
}

public function calculaDescuento(){
    return $this->calculaSubtotal()*0.1;
}

public function calculaNeto(){
    return $this->calculaSubtotal()-$this->calculaDescuento();
}
```

La función **calculaSubtotal** permite determinar el monto subtotal de la venta, se debe tener en cuenta que el precio fue implementado con el método determinaPrecio y que para referenciarlo debemos usar el operador **$this** como si se tratara de un atributo de clase.

Asimismo, la función **calculaDescuento** permite determinar el 10 % de descuento sobre el subtotal, tener en cuenta que se debe hacer referencia al método calculaSubtotal con el operador **$this**.

Finalmente, se implementa el método calculaNeto que permite restar el descuento del monto subtotal, de forma que ambos métodos deben ser referidos con el operador **$this**.

Archivo: **index.php**

```
<?php
    //Asociando el formulario a la clase Producto
    include('producto.php');
    if (isset($_POST['btnProcesar'])){
        $producto = new producto();
        $producto->setDescripcion($_POST['selProducto']);
        $producto->setCantidad($_POST['txtCantidad']);
    }

    //Función que mantiene el valor del producto seleccionado
    //en el cuadro combinado
    function productoxDefecto($producto){
        if ($producto=='Videograbadora') $selV='SELECTED'; else $selV='';
        if ($producto=='Camara') $selC='SELECTED'; else $selC='';
        if ($producto=='Televisor') $selT='SELECTED'; else $selT='';
        if ($producto=='Radiograbadora') $selR='SELECTED'; else $selR='';
        return array($selV,$selC,$selT,$selR);
    }
?>
<!DOCTYPE html>
<html>
```

```
<head>
  <meta charset="UTF-8">
  <link href="miEstilo.css" rel="stylesheet">
  <title>Venta de producto</title>
</head>
<body>
  <header>
     <h3>VENTA DE PRODUCTOS</h3>
     <h5>MANEJO DE CLASE-OBJETO CON PHP</h5>
  </header>
  <section>
    <form method="POST" action="index.php">
       <table border="0" width="600" cellspacing="5" cellpadding="0">
         <tr>
            <th>PRODUCTO</th>
            <?php
               if (isset($_POST['btnProcesar']))
                 lista($videograbadora,$camara,$televisor,$radiograbadora)=
                                  productoxDefecto($_POST['selProducto']);
            ?>
            <td><select name="selProducto">
                   <option value="Videograbadora"
                       <?php if (isset($_POST['btnProcesar']))
                          echo $videograbadora;?>>Videograbadora</option>
                   <option value="Camara"
                       <?php if (isset($_POST['btnProcesar']))
                          echo $camara; ?>>Camara fotográfica</option>
                   <option value="Televisor"
                       <?php if (isset($_POST['btnProcesar']))
                          echo $televisor; ?>>Televisor 32p</option>
                   <option value="Radiograbadora"
                       <?php if (isset($_POST['btnProcesar']))
                          echo $radiograbadora; ?>>Radiograbadora</option>
                </select>
              </td>
              <td rowspan="6"><img src="carrito.jpg"
                                 width="200" height="200" alt="carrito"/>
              </td>
         </tr>
         <tr>
            <th>CANTIDAD</th>
            <td>
               <input type="text" name="txtCantidad"
                     value="<?php
                                 if (isset($_POST['btnProcesar']))
                                    echo $_POST['txtCantidad'];
                            ?>" />
               <input type="submit" name="btnProcesar" value="PROCESAR" />
            </td>
          </tr>
          <tr>
            <th>SUBTOTAL</th>
            <td><?php
                if (isset($_POST['btnProcesar']))
                 echo("$ ".number_format($producto->calculaSubtotal(),'2'))
                 ?>
             </td>
           </tr>
           <tr>
              <th>DESCUENTO</th>
              <td><?php
                 if (isset($_POST['btnProcesar']))
                 echo("$ ".number_format($producto->calculaDescuento(),'2'))
                ?>
```

```
                </td>
              </tr>
              <tr>
                <th>NETO A PAGAR</th>
                <td><?php
                        if (isset($_POST['btnProcesar']))
                        echo("$ ".number_format($producto->calculaNeto(),'2'))
                    ?>
                </td>
              </tr>
            </table>
        </form>
        </section>
        <footer>
             <h5>Todos los derechos reservados a Lic. Manuel Torres</h5>
        </footer>
    </body>
</html>
```

Comentarios

```
<?php
    //Asociando el formulario a la clase Producto
    include('producto.php');
    if (isset($_POST['btnProcesar'])){
        $producto = new producto();
        $producto->setDescripcion($_POST['selProducto']);
        $producto->setCantidad($_POST['txtCantidad']);
    }

    //Función que mantiene el valor del producto seleccionado
    //en el cuadro combinado
    function productoxDefecto($producto){
        if ($producto=='Videograbadora') $selV='SELECTED'; else $selV='';
        if ($producto=='Camara') $selC='SELECTED'; else $selC='';
        if ($producto=='Televisor') $selT='SELECTED'; else $selT='';
        if ($producto=='Radiograbadora') $selR='SELECTED'; else $selR='';
        return array($selV,$selC,$selT,$selR);
    }
?>
```

La función include permite incluir la funcionalidad de la clase producto en el archivo **index.php**, con esto se puede tener acceso a todos los métodos públicos de la clase Producto.

Antes de crear un objeto de la clase, se evalúa si el usuario ha presionado el botón **btnProcesar** con el código (if(isset($_POST['btnProcesar']))); esto se hace con el fin de evitar errores de advertencia por parte del compilador PHP.

Luego se procede a crear un objeto de la clase producto, para esto se usa el operador new de la siguiente manera:

```
$producto  =new producto();
```

Donde **$producto** es el nombre del objeto y **producto()** hace referencia al método constructor de la clase Producto. Seguidamente se envía los valores ingresados en el formulario a los atributos privados de la clase, como recordará esto se realiza mediante la invocación a los métodos SET de la clase.

La función **productoxDefecto** tiene la misión de mantener el producto seleccionado por el usuario dentro del control cuadro combinado, mediante el parámetro **$producto** determinamos qué producto seleccionó el usuario y por medio de una comparación verificamos qué producto es para asignar la propiedad **SELECTED** a la opción seleccionada. Toda esta información es almacenada en grupo en un arreglo, eso quiere decir que la función **productoxDefecto** devuelve un arreglo de elementos que luego será desplegado en el control cuadro combinado.

```
<?php
    if (isset($_POST['btnProcesar']))
       lista($videograbadora,$camara,$televisor,$radiograbadora)=
                                    productoxDefecto($_POST['selProducto']);
?>
```

Para descomponer el arreglo en variables entendibles por el control cuadro combinado, se usará la función lista; tener en cuenta que el orden de asignación es exactamente igual al que se construyó en el arreglo.

```
<select name="selProducto">
    <option value="Videograbadora"
       <?php if (isset($_POST['btnProcesar']))
                 echo $videograbadora;?>>Videograbadora</option>
    <option value="Camara"
       <?php if (isset($_POST['btnProcesar']))
                 echo $camara; ?>>Camara fotográfica</option>
    <option value="Televisor"
       <?php if (isset($_POST['btnProcesar']))
                 echo $televisor; ?>>Televisor 32p</option>
    <option value="Radiograbadora"
       <?php if (isset($_POST['btnProcesar']))
                 echo $radiograbadora; ?>>Radiograbadora</option>
</select>
```

Una vez descompuesto los valores desde el arreglo, se imprimirán dichas variables dentro de la etiqueta del cuadro combinado.

```
<input type=”text” name=”txtCantidad”
       value=”<?php
                 if (isset($_POST[‘btnProcesar’]))
                       echo $_POST[‘txtCantidad’];
             ?>” />
```

La caja de texto **txtCantidad** también debe mantener el valor ingresado por el usuario, por tal motivo, se imprime en su propiedad value el valor capturado con $_POST['txtCantidad'].

```
<tr>
    <th>SUBTOTAL</th>
    <td><?php
            if (isset($_POST['btnProcesar']))
                 echo("$ ".number_format($producto->calculaSubtotal(),'2'))
        ?>
    </td>
</tr>
```

```
<tr>
    <th>DESCUENTO</th>
    <td><?php
             if (isset($_POST['btnProcesar']))
                 echo("$ ".number_format($producto->calculaDescuento(),'2'))
        ?>
    </td>
</tr>
<tr>
    <th>NETO A PAGAR</th>
    <td><?php
            if (isset($_POST['btnProcesar']))
                echo("$ ".number_format($producto->calculaNeto(),'2'))
        ?>
    </td>
</tr>
```

Finalmente, se imprimen los valores subtotal, descuento y neto implementados en la clase Producto, estos valores serán enviados a sus celdas correspondientes, se usa la función **number_format** para redondear a dos decimales los valores. Recuerde que para invocar un método de la clase, debe anteponer el nombre del objeto ($producto) seguido del operador -> y, por último, especificar el método.

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
#centrado{
    text-align: center;
}
table {
    margin: auto;
}
img{
    margin: auto;
    display: block;
}
table th {
    background-color: #b9c9fe;
    text-align: left;
}
```

Caso desarrollado 2 **Método constructor-Pago de trabajadores**

Implementar una aplicación web con PHP que permita determinar el salario neto de los trabajadores usando método constructor, tal como se muestra en la siguiente imagen:

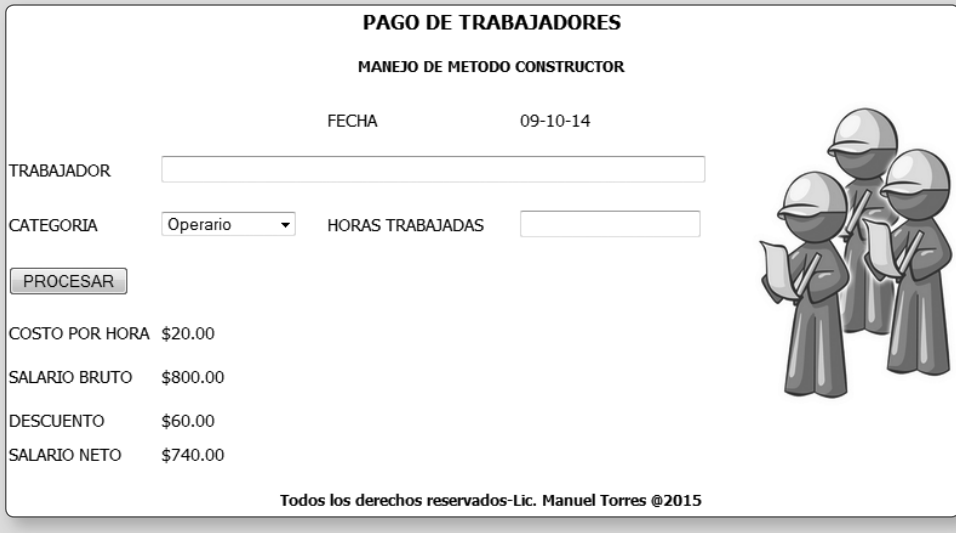


Imagen: index.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **metodoConstructor**.
- Al iniciar la aplicación deberá mostrar la página **index.php**.
- También tener en cuenta los siguientes aspectos:

  - Mostrar automáticamente la fecha de registro, este se obtiene desde el sistema operativo.
  - Mostrar las siguientes categorías en el control cuadro combinado.
  - Operario
  - Administrativo
  - Jefe

- Al presionar el botón **PROCESAR** se debe mantener tanto el nombre, la categoría y las horas laboradas por el trabajador, además de mostrar el costo por hora, salario bruto, descuento y salario neto.

El objetivo de cada archivo que contiene el proyecto es:

- **pago.php**: Es considerado la clase pago, el cual contiene atributos, método constructor, métodos GET y SET, y los métodos propios de la aplicación, como determinar el costo por hora, calcular el subtotal, el descuento y el neto a pagar.
- **index.php**: Contiene la página web que interactúa con el usuario para gestionar el pago de los trabajadores.
- **miEstilo.css**: Contiene los estilos usados en el archivo **index.php**.

Archivo: **pago.php**

```
<?php
    class pago {
        //Atributos de clase
        private $fecha='';
        private $trabajador='';
        private $categoria='';
        private $horas=0;

        //Método constructor
        public function __construct($fecha,$trabajador,$categoria,$horas) {
            $this->fecha=$fecha;
            $this->trabajador=$trabajador;
            $this->categoria=$categoria;
            $this->horas=$horas;
        }

        //Methods GET
        public function getFecha(){
            return $this->fecha;
        }
        public function getTrabajador(){
            return $this->trabajador;
        }
        public function getCategoria(){
            return $this->categoria;
        }
        public function getHoras(){
            return $this->horas;
        }

        //Métodos SET
        public function setFecha($fecha){
            $this->fecha=$fecha;
        }
        public function setTrabajador($trabajador){
            $this->trabajador=$trabajador;
        }
        public function setCategoria($categoria){
            $this->categoria=$categoria;
        }
        public function setHoras($horas){
            $this->horas=$horas;
        }

        //Métodos propios
        public function determinaCostoHora(){
        if ($this->categoria=='Operario')
            return 10;
            elseif ($this->categoria=='Administrativo')
            return 20;
            elseif ($this->categoria=='Jefe')
            return 40;
            else
                return 0;
        }

        public function calculaSubtotal(){
            return $this->determinaCostoHora()*$this->horas;
        }
```

```
        public function calculaDescuento(){
             if ($this->calculaSubtotal()<1000)
                  return 0.075*$this->calculaSubtotal();
                elseif ($this->calculaSubtotal()<=2000)
                  return 0.14*$this->calculaSubtotal();
                elseif ($this->calculaSubtotal()>2000)
                  return 0.20*$this->calculaSubtotal();
                else
                  return 0;
          }

        public function calculaNeto(){
              return $this->calculaSubtotal()-$this->calculaDescuento();
        }
    }
?>
```

Comentarios

```
private $descripcion='';
private $cantidad=0;
```

Declaramos los atributos de clase para la descripción del producto ($descripcion) y la cantidad comprada por el cliente ($cantidad), no se declara el precio del producto ya que existe un método que asignará los precios según el producto seleccionado. Ambos atributos han sido inicializados según la naturaleza de sus datos. Recuerde que los atributos de clase son declarados como privados.

```
public function determinaCostoHora(){
   if ($this->categoria=='Operario')
      return 10;
      elseif ($this->categoria=='Administrativo')
      return 20;
      elseif ($this->categoria=='Jefe')
      return 40;
      else
      return 0;
   }

public function calculaSubtotal(){
   return $this->determinaCostoHora()*$this->horas;
}

public function calculaDescuento(){
   if ($this->calculaSubtotal()<1000)
      return 0.075*$this->calculaSubtotal();
      elseif ($this->calculaSubtotal()<=2000)
      return 0.14*$this->calculaSubtotal();
      elseif ($this->calculaSubtotal()>2000)
      return 0.20*$this->calculaSubtotal();
      else
      return 0;
}

public function calculaNeto(){
    return $this->calculaSubtotal()-$this->calculaDescuento();
}
```

La función **determinaCostoHora** tiene la misión de establecer un determinado costo según la categoría seleccionado por el usuario, se debe considerar que $this->categoría es la forma adecuada de poder invocar a los atributos privados de la clase.

Por otra parte, se debe calcular el subtotal a pagar, el cual se determina multiplicando las horas trabajadas por el costo por hora. A su vez, se calcula el descuento a partir del valor emitido por la función **determinaSubtotal** invocado como $this->determinaSubtotal, ya que es una función de clase. Finalmente, se calcula el monto neto, el cual resulta del residuo entre el subtotal y el descuento.

Archivo: **index.php**

```
<?php
    include('pago.php');
    if (isset($_POST['btnProcesar'])){
        $fecha = date('d-m-y');
        $trabajador = $_POST['txtTrabajador'];
        $categoria = $_POST['selCategoria'];
        $horas = $_POST['txtHoras'];

        $objPago = new pago($fecha,$trabajador,$categoria,$horas);
    }
?>
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <link href="miEstilo.css" rel="stylesheet">
  </head>
  <body>
    <header>
      <h3>PAGO DE TRABAJADORES</h3>
      <h5>MANEJO DE METODO CONSTRUCTOR</h5>
    </header>
    <section>
      <form method="POST">
        <table border="0" width="800" cellspacing="1" cellpadding="1">
          <tr>
            <td></td>
            <td></td>
            <td>FECHA</td>
            <td><?php echo date('d-m-y'); ?></td>
            <td rowspan="7">
                 <img src="trabajador.png" width="180" height="280" />
            </td>
          </tr>
          <tr>
             <td>TRABAJADOR</td>
             <td colspan="3">
                   <input type="text" name="txtTrabajador"
                          size="70" value="" />
             </td>
          </tr>
          <tr>
             <td>CATEGORIA</td>
             <td><select name="selCategoria">
                    <option value="Operario">Operario</option>
                    <option value="Administrativo">Administrativo</option>
```

```
                    <option value="Jefe">Jefe</option>
                  </select>
              </td>
              <td>HORAS TRABAJADAS</td>
              <td><input type="text" name="txtHoras" value="" /></td>
            </tr>
            <tr>
              <td>
                <input type="submit" value="PROCESAR" name="btnProcesar"/>
              </td>
              <td></td>
              <td></td>
              <td></td>
            </tr>
            <tr>
              <td>COSTO POR HORA</td>
              <td><?php if (isset($_POST['btnProcesar']))
                    echo '$'.number_format($objPago->determinaCostoHora(),2);
                  ?>
              </td>
              <td></td>
              <td></td>
            </tr>
            <tr>
              <td>SALARIO BRUTO</td>
              <td><?php if (isset($_POST['btnProcesar']))
                        echo '$'.number_format($objPago->calculaSubtotal(),2);
                  ?>
              </td>
              <td></td>
              <td></td>
            </tr>
            <tr>
              <td>DESCUENTO</td>
              <td><?php if (isset($_POST['btnProcesar']))
                        echo '$'.number_format($objPago->calculaDescuento(),2);
                  ?>
              </td>
              <td></td>
              <td></td>
            </tr>
            <tr>
              <td>SALARIO NETO</td>
              <td><?php if (isset($_POST['btnProcesar']))
                         echo '$'.number_format($objPago->calculaNeto(),2);
                  ?>
              </td><td></td>
              <td></td>
            </tr>
          </table>
      </section>
      <footer>
         <h5>Todos los derechos reservados-Lic. Manuel Torres @2015</h5>
      </footer>
    </form>
  </body>
</html>
```

Comentarios

```
<?php
    include('pago.php');
    if (isset($_POST['btnProcesar'])){
        $fecha = date('d-m-y');
        $trabajador = $_POST['txtTrabajador'];
        $categoria = $_POST['selCategoria'];
        $horas = $_POST['txtHoras'];

        $objPago = new pago($fecha,$trabajador,$categoria,$horas);
    }
?>
```

Se incluye el archivo **pago.php** para poder hacer uso de todos los métodos y atributos públicos de la clase pago. Seguidamente, se evalúa si el usuario ha seleccionado el botón **btnProcesar**, luego se captura todos los valores necesarios para el constructor de la clase pago, como la fecha de registro, nombre del trabajador, categoría y las horas trabajadas.

Finalmente, se crea un objeto llamado $objPago, de la clase pago, y se envían todos los valores capturados en estricto orden, este orden se propuso al momento de implementar el método constructor en la clase pago.

```
<tr>
  <td></td>
  <td></td>
  <td>FECHA</td>
  <td><?php echo date('d-m-y'); ?></td>
  <td rowspan="7">
     <img src="trabajador.png" width="180" height="280" />
  </td>
</tr>
```

Para imprimir la fecha actual se debe invocar al método date, especificando como parámetro la forma en que se mostrará la fecha «d-m-y» (día, mes y año).

```
<tr>
  <td>COSTO POR HORA</td>
  <td><?php if (isset($_POST['btnProcesar']))
                  echo '$'.number_format($objPago->determinaCostoHora(),2);
      ?>
  </td>
  <td></td>
  <td></td>
</tr>
```

Se muestra el costo por hora calculado en la clase pago por medio de la función determinaCostoHora, el cual debe ser invocado por medio del objeto objPago. Como complemento se usa la función number_format para redondear el resultado a dos decimales.

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
#centrado{
    text-align: center;
}
table {
    margin: auto;
}
img{
    margin: auto;
    display: block;
}
table th {
    background-color: #b9c9fe;
    text-align: left;
}
```

Caso desarrollado 3 Método constructor-Listado de productos con MySQL

Implementar una aplicación web con PHP que permita listar los productos desde la base de datos VENTAS2015, usando el método constructor desde la clase Producto, tal como se muestra en la siguiente imagen:

LISTADO DE PRODUCTOS

USANDO MÉTODO CONSTRUCTOR

CÓDIGO	DESCRIPCIÓN	PRECIO	STOCK	FECHA VENC.
P0001	PYE DE MANZANA I	20.00	500	2015-05-14
P0002	TORTA DE CHOCOLATE	45.00	100	2015-07-04
P0003	TORTA TRES LECHES	30.00	40	2015-06-24
P0004	MOUSE DE MANZANA	35.00	70	2015-09-06
P0005	ARROZ CON LECHE-ENVASE ESPECIAL	13.00	40	2015-11-04
P0006	MAZAMORRA MORADA	1.50	70	2015-12-04
P0007	YOGUR ARABE	3.00	100	2015-05-04
P0008	PAN CON POLLO	2.00	500	2015-06-05
P0009	BROWNIE	3.00	300	2015-05-04
P0010	BESO DE MOZA	1.00	400	2015-06-03
P0011	PYE DE LIMON	1.70	100	2015-05-02
P0012	TORTA DE NARANJA	16.00	10	2015-04-07
P0013	TORTA DE FRESA	41.00	100	2015-05-07
P0014	ALFAJORES	0.30	400	2015-04-06
P0015	CHOCOTEJAS	2.00	100	2015-04-14
P0016	SUSPIRO A LA LIMEÑA	3.50	100	2015-06-12

EL TOTAL DE PRODUCTOS:16

Todos los derechos reservados @2015

Imagen: index.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **ListadoProductosConstructor**.
- Al iniciar la aplicación deberá mostrar la página **index.php**, la cual contiene una lista de todos los productos desde la base de datos Ventas2015.
- Al final de la página se debe mostrar el total de productos listados.

El objetivo de cada archivo que contiene el proyecto es:

- **conexion.php**: Es la clase que contiene los métodos principales del listado de productos:
  - Método constructor que inicializa la cadena de conexión a la base de datos Ventas2015.
  - Función que permita ejecutar la consulta de tipo SELECT y que permita almacenar el resultado en un arreglo de productos.
  - Función que permita mostrar el número total de productos a partir de la consulta de tipo SELECT.
- **index.php**: Contiene la página web que interactúa con el usuario para listar los productos a partir de los métodos implementados en la clase **conexión.php**.
- **miEstilo.css**: Contiene los estilos usados en el archivo **index.php**.

Archivo: **conexion.php**

```
<?php
    class conexion{
        private $cn=null;

        //Método constructor que permite conectarse a la base de datos
        public function __construct() {
            $this->cn=mysql_connect('localhost','root','');
            mysql_select_db('Ventas2015',$this->cn);
        }

        //Creando un arreglo de los productos
        public function consulta($sql){
            $rs=mysql_query($sql,  $this->cn);
            while($misProductos=mysql_fetch_array($rs)){
                $productos[]=$misProductos;
            }
            return $productos;
        }

        //Determinar el total de productos registrados
        public function totalProductos($sql){
            $rs=mysql_query($sql,$this->cn);
            return mysql_num_rows($rs);
        }
    }
?>
```

Comentarios

```
public function __construct() {
  $this->cn=mysql_connect('localhost','root','');
  mysql_select_db('Ventas2015',$this->cn);
}
```

El método constructor permite implementar la conexión al servidor por medio de la variable **$this->cn** y la selección de la base de datos Ventas2015 mediante la función **mysql_select_db**, se debe considerar que un método constructor no devuelve valor alguno, es así que esta función no tiene return, así como la función **mysql_select_db** tampoco tiene una variable asociada.

```
public function consulta($sql){
   $rs=mysql_query($sql,  $this->cn);
   while($misProductos=mysql_fetch_array($rs)){
       $productos[]=$misProductos;
   }
   return $productos;
}
```

La función **consulta** permite obtener todos los registros a partir de una sentencia recibida como parámetro; para este caso, se lo recibe en el parámetro **$sql**. Seguidamente, se recorre todos los registros con la estructura **while**, para luego ser enviado a una variable de tipo arreglo, es justamente esta variable la que debe ser devuelta por la función, usando la sentencia return **$productos**.

```
public function totalProductos($sql){
   $rs=mysql_query($sql,$this->cn);
   return mysql_num_rows($rs);
}
```

La función **totalProductos** es la encargada de determinar el total de productos encontrados en una determinada consulta, la cual se envía mediante el parámetro **$sql**.

Archivo: **index.php**

```
<!DOCTYPE html>
<html>
 <head>
   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
   <title></title>
   <link href="miEstilo.css" rel="stylesheet">
 </head>
 <body>
   <header>
      <img src="banner.jpg" width="700" height="170"/>
      <h3>LISTADO DE PRODUCTOS</h3>
      <h5>USANDO METODO CONSTRUCTOR</h5>
   </header>
   <section>
     <?php
        include('conexion.php');
        $objCon = new conexion();
        $sql = 'SELECT ID_PRODUCTO,DESCRIPCION,PRECIO,STOCK,FECHA_VENC
```

```
                        FROM PRODUCTO';
          $productos=$objCon->consulta($sql);
       ?>
       <table border="0" width="650" cellspacing="0" cellpadding="0">
         <tr>
           <th>CODIGO</th>
           <th>DESCRIPCION</th>
           <th>PRECIO</th>
           <th>STOCK</th>
           <th>FECHA VENC.</th>
         </tr>
         <?php
          foreach ($productos as $prod){ ?>
         <tr>
           <td><?php echo $prod['id_producto']; ?></td>
           <td><?php echo utf8_decode($prod['descripcion']); ?></td>
           <td><?php echo $prod['precio']; ?></td>
           <td><?php echo $prod['stock']; ?></td>
           <td><?php echo $prod['fecha_venc']; ?></td>
          </tr>
          <?php } ?>
          <tr>
              <th colspan="5">
                 <?php echo 'EL TOTAL DE PRODUCTOS:'
                                  .$objCon->totalProductos($sql); ?>
              </th>
          </tr>
          </table>
          </section>
          <footer>
              <h5>Todos los derechos reservados @2015</h5>
          </footer>
     </body>
</html>
```

Comentarios

```
<?php
  include('conexion.php');
  $objCon = new conexion();
  $sql = 'SELECT ID_PRODUCTO,DESCRIPCION,PRECIO,STOCK,FECHA_VENC
                  FROM PRODUCTO';
  $productos=$objCon->consulta($sql);
?>
```

Se incluye al archivo **index.php** las variables de conexión al servidor(cn) y acceso a la base de datos Ventas2015, luego se crea un objeto de la clase conexión; esto permitirá acceder a los métodos implementados dentro de la clase conexión, tener en cuenta que dichos métodos deben tener visibilidad pública.

Luego, se asigna el script de consulta a la variable **$sql** que se enviará luego a la función **consulta** y, finalmente, la variable **$productos** recibe la información obtenida desde la ejecución de la consulta $objCon->consulta($sql).

```
<?php
   foreach ($productos as $prod){ ?>
   <tr>
     <td><?php echo $prod['id_producto']; ?></td>
     <td><?php echo utf8_decode($prod['descripcion']); ?></td>
     <td><?php echo $prod['precio']; ?></td>
     <td><?php echo $prod['stock']; ?></td>
     <td><?php echo $prod['fecha_venc']; ?></td>
   </tr>
<?php } ?>
```

Para poder imprimir los registros, se usa la estructura repetitiva **foreach**, la cual envía toda la información almacenada en el arreglo **$productos** a la variable **$prod**; y es aquí de donde se obtendrán todos los valores registrados en el arreglo y enviados a las celdas correspondientes. Finalmente, se debe tener en cuenta que para referenciar a una columna se debe usar el siguiente formato $prod['id_producto'] o en su defecto se podría usar el número de columna, es decir, también sería válido usar $prod[0].

```
<tr>
  <th colspan="5">
    <?php echo 'EL TOTAL DE PRODUCTOS:'.$objCon->totalProductos($sql); ?>
  </th>
</tr>
```

Recuerde que en la clase **conexión.php** se implementó un método llamado **totalProductos**, el cual solicitaba un parámetro para precisar la cantidad de registros a partir de una determinada consulta, la forma de invocar al método se realizará mediante el objeto asociado a la clase conexión($objCon).

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
table {
    margin: auto;
}
img{
    margin: auto;
    display: block;
}
table th {
    background-color: #b9c9fe;
    text-align: left;
}
```

Caso desarrollado 4 **Mantenimiento de productos orientado a objetos**

Implementar una aplicación web con PHP que permita gestionar el mantenimiento de registros de la tabla Producto, usando programación orientada a objetos desde la base de datos VENTAS2015, tal como se muestra en la siguiente imagen:

CÓDIGO	DESCRIPCIÓN	PRECIO	STOCK	FECHA	ACCIONES
P0001	PYE DE MANZANA I	20.00	500	2015-05-14	
P0002	TORTA DE CHOCOLATE	45.00	100	2015-07-04	
P0003	TORTA TRES LECHES	30.00	40	2015-06-24	
P0004	MOUSE DE MANZANA	35.00	70	2015-09-06	
P0005	ARROZ CON LECHE-ENVASE ESPECIAL	13.00	40	2015-11-04	
P0006	MAZAMORRA MORADA	1.50	70	2015-12-04	
P0007	YOGURT ARABE	3.00	100	2015-05-04	
P0008	PAN CON POLLO	2.00	500	2015-06-05	
P0009	BROWNIE	3.00	300	2015-05-04	
P0010	BESO DE MOZA	1.00	400	2015-06-03	
P0011	PYE DE LIMON	1.70	100	2015-05-02	
P0012	TORTA DE NARANJA	16.00	10	2015-04-07	
P0013	TORTA DE FRESA	41.00	100	2015-05-07	
P0014	ALFAJORES	0.30	400	2015-04-06	
P0015	CHOCOTEJAS	2.00	100	2015-04-14	
P0016	SUSPIRO	3.50	100	2015-06-12	

Imagen: index.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **mantenimientoProductosOO**.
- Se cuenta con una carpeta de imágenes dentro del proyecto llamado imágenes:

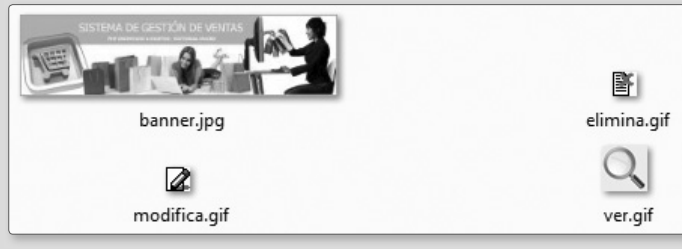


Imagen: contenido de la carpeta imágenes

- Al iniciar la aplicación deberá mostrar la página **index.php**, la cual mostrará una lista de los productos registrados en la base de datos, además de las acciones que puede realizar sobre dichos registros como los siguientes:

  - Visualizar los datos en una hoja independiente.
  - Editar o modificar los datos del producto.
  - Eliminar o borrar el registro de producto.

- Para añadir un nuevo producto en la tabla, se realiza mediante el botón **Agregar nuevo Producto,** el cual muestra la siguiente interface:

Imagen: nuevo.php

Se debe tener en cuenta que el código del producto nuevo debe ser autogenerado, además de estar protegido de cambios por parte del usuario y que cada control usado en la interfaz del nuevo producto debe indicar lo que el usuario debe ingresar; por último, al ingresar los datos del nuevo producto se debe guardar con el botón **REGISTRAR** para, finalmente, mostrar un mensaje de «Producto agregado correctamente».

Imagen: index.php

- Para listar los productos registrados en la tabla Producto, se debe presionar el botón **LISTADO DE PRODUCTOS**, el aspecto será similar al mostrado inicialmente, es decir, listará los productos además del nuevo producto registrado.
- A partir del listado de productos, se pueden realizar diferentes acciones, como visualizar, modificar y/o actualizar o eliminar un determinado producto. Al seleccionar la opción , se pueden visualizar los datos de un determinado producto, tal como se muestra en la siguiente interface:

Imagen: visualiza.php

Al seleccionar la opción , se pueden visualizar los datos de un determinado producto, a partir de aquí es posible modificar algún dato perteneciente al producto, tener en cuenta que el código mostrado no puede ser modificado.

Imagen: actualiza.php

Al realizar la modificación de algún valor del producto, se deberán registrar dichos cambios presionando el botón **ACTUALIZA** y automáticamente se mostrará el listado de los productos.

Al seleccionar la opción se podrá eliminar los datos de un determinado producto, a partir de aquí se puede eliminar al producto.

Imagen: elimina.php

El objetivo de cada archivo que contiene el proyecto es:

- **encabezado.php:** Contiene el banner y los botones de **AGREGAR NUEVO PRODUCTO** y **LISTADO DE PRODUCTOS**, que se mostrarán en todos los documentos web del proyecto.
- **producto. Php:** Contiene la especificación de la clase Producto, el cual contiene métodos GET y SET para todos los atributos.
- **productoCN.php:** Contiene los métodos que permiten consultar, registrar, actualizar, eliminar y generar el código de un nuevo producto.
- **conexion.php:** Es la clase que contiene los siguientes métodos:
  - Método constructor que inicializa la cadena de conexión y accede a la base de datos Ventas2015.
  - Función que permite devolver la conexión implementada en el constructor.
- **index.php:** Es la página principal del proyecto, es desde aquí donde el usuario podrá desplazarse por el proyecto.
- **listado.php:** Permite listar los registros de los productos, incluyendo los botones de visualización, actualización y eliminación de los registros.
- **visualiza.php:** Permite obtener el código del producto desde el listado y muestra la información del mismo.
- **nuevo.php:** Permite registrar los datos del nuevo producto, aquí se debe considerar que el código debe mostrarse de forma automática.
- **actualiza.php:** Permite obtener el código del producto a partir del listado y a la vez actualiza los valores del mismo.
- **elimina.php**: Obtiene el código del producto a partir del listado y a la vez elimina dicho registro.
- **cerrar.php**: Permite cerrar la conexión establecida en el archivo **conexión.php**.
- **estilo.css**: Contiene los estilos usados en todos los archivos del proyecto.

Archivo: **encabezado.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
  </head>
  <body>
  <header>
    <img src="imagenes/banner.jpg" width="750" id="banner"/>
    <h3>MANTENIMIENTO DE PRODUCTOS</h3>
  </header>
  <section>
    <form method="POST">
      <table border="0" width="750" cellspacing="1" cellpadding="1">
        <tr>
          <td colspan="6">
            <input type="submit" value="AGREGAR NUEVO PRODUCTO"
                                 name="btnMantenimiento"/> |
            <input type="submit" value="LISTADO DE PRODUCTOS"
                                 name="btnMantenimiento"/>
          </td>
         </tr>
      </table>
    </form>
  </section>
  </body>
</html>
```

Archivo: **producto.php**

```
<?php
    class producto{
        private $código='';
        private $descripcion='';
        private $precio=0;
        private $stock=0;
        private $fecha='';

        //Implementación de los métodos SET
        public function setCodigo($código){
            $this->código=$código;
        }
        public function setDescripcion($descripcion){
            $this->descripcion=$descripcion;
        }
        public function setPrecio($precio){
            $this->precio=$precio;
        }
        public function setStock($stock){
            $this->stock=$stock;
        }
        public function setFecha($fecha){
            $this->fecha=$fecha;
        }

        //Implementación de los métodos GET
        public function getCodigo(){
            return $this->codigo;
        }
```

```
        public function getDescripcion(){
            return $this->descripcion;
        }
        public function getPrecio(){
            return $this->precio;
        }
        public function getStock(){
            return $this->stock;
        }
        public function getFecha(){
            return $this->fecha;
        }
    }
?>
```

Archivo: **productoCN.php**

```
<?php
    class productoCN{

    //Función que permite listar los productos
    public function consulta($sql){
      include('conexion.php');
      $objCon = new conexion();
      $cn=$objCon->getConecta();
      $rs=mysql_query($sql,  $cn);
      while($misProductos=mysql_fetch_array($rs)){
         $productos[]=$misProductos;
      }
      return $productos;
    }

    //Función que permite registrar un nuevo producto
    public function registra($cod,$desc,$pre,$sto,$fec){
      include('conexion.php');
      $objCon = new conexion();
      $cn=$objCon->getConecta();

      $rs=mysql_query("INSERT INTO PRODUCTO
                              VALUES ('$cod','$desc',$pre,$sto,'$fec')",$cn);
      if ($rs)
        echo "<h5>Registro de producto correcto..!!</h5>";
        else
        echo "<h5>Ocurrió un error al registrar el producto "
                                                  .mysql_error()."</h5>";
    }

    //Método que permite generar código nuevo de producto
    public function generaCodigo(){
      include('conexion.php');
      $objCon = new conexion();
      $cn=$objCon->getConecta();

      $rs=mysql_query("SELECT ID_PRODUCTO FROM PRODUCTO
                               ORDER BY ID_PRODUCTO DESC LIMIT 1;",$cn);
        $fila = mysql_fetch_row($rs);
        return 'P'.str_pad((substr($fila[0], -4)+1),4,'0',STR_PAD_LEFT);;
    }

   //Método que permite buscar un determimado producto por su código
   public function buscaProducto($código){
        include('conexion.php');
```

```
        $objCon = new conexion();
        $cn=$objCon->getConecta();

        $rs=mysql_query("SELECT * FROM PRODUCTO
                                WHERE ID_PRODUCTO='$codigo'",$cn);
        $miProducto = mysql_fetch_array($rs);
        return $miProducto;
    }

    //Método que permite actualizar un registro de producto
     public function actualiza($cod,$desc,$pre,$sto,$fec){
        include('conexion.php');
        $objCon = new conexion();
        $cn=$objCon->getConecta();

        $rs=mysql_query("UPDATE PRODUCTO SET DESCRIPCION='$desc', PRECIO=$pre,
                                STOCK=$sto, FECHA_VENC='$fec'
                                WHERE ID_PRODUCTO='$cod'",$cn);
        if ($rs)
            echo "<h5>Producto actualizado correctamente..!!</h5>";
            else
            echo "<h5>Ocurrió un error al actualizar el producto "
                 .  mysql_error()."</h5>";
     }

     //Método que permite eliminar un determinado producto
     public function elimina($cod){
        include('conexion.php');
        $objCon = new conexion();
        $cn=$objCon->getConecta();

        $rs=mysql_query("DELETE FROM PRODUCTO WHERE ID_PRODUCTO='$cod'",$cn);
        if ($rs)
            echo "<h5>Producto eliminado correctamente..!!</h5>";
       else
          echo "<h5>Ocurrió un error al eliminar el producto "
             . mysql_error()."</h5>";
     }
    }
  ?>
```

Archivo: **conexion.php**

```
<?php
    class conexion{
        private $cn=null;

        //Método constructor que permite conectarse a la base de datos
        public function __construct() {
            $this->cn=mysql_connect('localhost','root','');
            mysql_select_db('Ventas2015',$this->cn);
        }

        public function getConecta(){
            return $this->cn;
        }
    }
?>
```

Archivo: **index.php**

```
<!DOCTYPE html>
<html>
  <head>
        <meta charset="UTF-8">
        <title></title>
        <link href="estilo.css" rel="stylesheet">
  </head>
  <body>
    <?php
      error_reporting(0);
      include('encabezado.php');

    //Verificamos qué botón seleccionó el usuario (Nuevo, Listado o
    //Registrar)
      if (isset($_POST['btnMantenimiento'])){
         $botón=$_POST['btnMantenimiento'];
         if ($botón=="AGREGAR NUEVO PRODUCTO"){
           include('nuevo.php');
           }elseif ($botón=="LISTADO DE PRODUCTOS"){
             include('listado.php');
           }elseif ($botón=="REGISTRAR"){
             include('producto. Php');
             include('productoCN.php');
             $objPro = new producto();
             $objNeg = new productoCN();

             $objPro->setCodigo($_POST['txtCodigo']);
             $objPro->setDescripcion($_POST['txtDescripcion']);
             $objPro->setPrecio($_POST['txtPrecio']);
             $objPro->setStock($_POST['txtStock']);
             $objPro->setFecha($_POST['txtFecha']);

             $objNeg->registra($objPro->getCodigo(),$objPro->getDescripcion(),
                               $objPro->getPrecio(),$objPro->getStock(),
                               $objPro->getFecha());
           }
       }
       else
         include('listado.php');
     ?>
</body>
</html>
```

Archivo: **listado.php**

```
<?php
    error_reporting(0);
    include('productoCN.php');

    //Consultando todos los registros de productos
    $objNeg = new productoCN();
    $sql = 'SELECT id_producto,descripcion,precio,stock,fecha_venc
                   FROM PRODUCTO';
    $productos=$objNeg->consulta($sql);
?>
<form method="POST">
<table border="0" width="750" cellspacing="1" cellpadding="1">
        <tr>
            <th>CODIGO</th>
            <th>DESCRIPCION</th>
```

```
            <th>PRECIO</th>
            <th>STOCK</th>
            <th>FECHA</th>
            <th>ACCIONES</th>
        </tr>
        <?php
        //Listando todos los productos
          foreach ($productos as $prod){
        ?>
        <tr>
         <td><?php echo $prod['id_producto']; ?></td>
         <td><?php echo utf8_decode($prod['descripcion']); ?></td>
         <td><?php echo $prod['precio']; ?></td>
         <td><?php echo $prod['stock']; ?></td>
         <td><?php echo $prod['fecha_venc']; ?></td>
         <td id="centrado">
             <a href="visualiza.php?id=<?php echo $prod['id_producto']; ?>">
                <img src="imagenes/ver.gif" width="20" height="20" />
             </a>
             <a href="actualiza.php?id=<?php echo $prod['id_producto']; ?>">
                 <img src="imagenes/modifica.gif" width="20" height="20"/>
             </a>
             <a href="elimina.php?id=<?php echo $prod['id_producto']; ?>">
             <img src="imagenes/elimina.gif" width="20" height="20" />
             </a>
         </td>

        </tr>
        <?php } ?>
</table>
</form>
```

Archivo: **visualiza.php**

```
<!DOCTYPE html>
<html>
        <meta charset="UTF-8">
        <title></title>
        <link href="estilo.css" rel="stylesheet">
    </head>
    <body>
        <?php
            error_reporting(0);
            include('encabezado.php');
            if (isset($_POST['btnMantenimiento'])){
                $botón=$_POST['btnMantenimiento'];
            if ($botón=="AGREGAR NUEVO PRODUCTO"){
                include('nuevo.php');
            }elseif ($botón=="LISTADO DE PRODUCTOS"){
                include('listado.php');
            }elseif ($botón=="REGISTRAR"){
                include('producto. Php');
                include('productoCN.php');
                $objPro = new producto();
                $objNeg = new productoCN();

                $objPro->setCodigo($_POST['txtCodigo']);
                $objPro->setDescripcion($_POST['txtDescripcion']);
                $objPro->setPrecio($_POST['txtPrecio']);
                $objPro->setStock($_POST['txtStock']);
                $objPro->setFecha($_POST['txtFecha']);
```

```
        $objNeg->registra($objPro->getCodigo(),$objPro->getDescripcion(),$objPro-
>getPrecio(),$objPro->getStock(),$objPro->getFecha());
        }
        }
         ?>

    <?php
      include('productoCN.php');
      $objNeg = new productoCN();

      //Obteniendo el código desde el listado
      $código=$_GET['id'];

      //Buscando el producto seleccionado
      $productos=$objNeg->buscaProducto($código);
    ?>

    <form method="POST">
    <table border="0" width="750" cellspacing="1" cellpadding="1">
    <tr>
       <th colspan="2">DESCRIPCIÓN DEL PRODUCTO</th>
    </tr>
    <tr>
        <td>CODIGO</td>
        <td><?php echo $productos[0]; ?></td>
    </tr>
    <tr>
        <td>DESCRIPCION</td>
        <td><?php echo $productos[1]; ?></td>
        <td></td>
    </tr>
    <tr>
        <td>PRECIO</td>
        <td><?php echo $productos[2]; ?></td>
        <td></td>
    </tr>
    <tr>
        <td>STOCK</td>
        <td><?php echo $productos[3]; ?></td>
        <td></td>
    </tr>
    <tr>
        <td>FECHA VENC.</td>
        <td><?php echo $productos[4]; ?></td>
        <td></td>
    </tr>
    </table>
</form>
</html>
```

Archivo: **nuevo.php**

```
<?php
    error_reporting(0);
    include('productoCN.php');
    $objNeg = new productoCN();

    //$cod contiene el código autogenerado del producto nuevo
    $cod=$objNeg->generaCodigo();
?>
```

```
<form method="POST">
   <table border="0" width="750"
              cellspacing="0" cellpadding="5">
   <tr>
       <th colspan="2">REGISTRO DEL NUEVO PRODUCTO</th>
   </tr>
   <tr>
      <td>CODIGO</td>
      <td><input type="text" name="txtCodigo"
                 value="<?php echo $cod; ?>" readonly /></td>
   </tr>
   <tr>
     <td>DESCRIPCION</td>
     <td><input type="text" name="txtDescripcion"
                size="70" value=""
                placeholder="Registre descripción del producto"/></td>
   </tr>
   <tr>
     <td>PRECIO</td>
     <td><input type="text" name="txtPrecio"
                value="" placeholder="Ingrese precio"/></td>
   </tr>
   <tr>
     <td>STOCK</td>
     <td><input type="text" name="txtStock"
                value="" placeholder="Ingrese stock"/></td>
   </tr>
   <tr>
     <td>FECHA DE VENC.</td>
     <td><input type="text" name="txtFecha"
                value="" placeholder="año/mes/día"/></td>
   </tr>
   <tr>
     <td></td>
     <td><input type="submit" name="btnMantenimiento"
                value="REGISTRAR" /></td>
   </tr>
  </table>
</form>
```

Archivo: **actualiza.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
    <link href="estilo.css" rel="stylesheet">
  </head>
  <body>
    <?php
      error_reporting(0);
      include('encabezado.php');
      if (isset($_POST['btnMantenimiento'])){
        $botón=$_POST['btnMantenimiento'];
        if ($botón=="AGREGAR NUEVO PRODUCTO"){
          include('nuevo.php');
        }elseif ($botón=="LISTADO DE PRODUCTOS"){
          include('listado.php');
```

```
        }elseif ($botón=="REGISTRAR"){
          include('producto. Php');
          include('productoCN.php');

          $objPro = new producto();
          $objNeg = new productoCN();

          $objPro->setCodigo($_POST['txtCodigo']);
          $objPro->setDescripcion($_POST['txtDescripcion']);
          $objPro->setPrecio($_POST['txtPrecio']);
          $objPro->setStock($_POST['txtStock']);
          $objPro->setFecha($_POST['txtFecha']);

          $objNeg->registra($objPro->getCodigo(),$objPro->getDescripcion(),
                            $objPro->getPrecio(),$objPro->getStock(),
                            $objPro->getFecha());
       }

       elseif ($boton=="ACTUALIZA"){
         include('producto.php');
         include('productoCN.php');
         $objPro = new producto();
         $objNeg = new productoCN();

         $objPro->setCodigo($_POST['txtCodigo']);
         $objPro->setDescripcion($_POST['txtDescripcion']);
         $objPro->setPrecio($_POST['txtPrecio']);
         $objPro->setStock($_POST['txtStock']);
         $objPro->setFecha($_POST['txtFecha']);

         $objNeg->actualiza($objPro->getCodigo(),$objPro->getDescripcion(),
                            $objPro->getPrecio(),$objPro->getStock(),
                            $objPro->getFecha());
       }
     }
   ?>

   <?php
     include('productoCN.php');
     $objNeg = new productoCN();

     $código=$_GET['id'];

     $productos=$objNeg->buscaProducto($código);
   ?>

     <form method="POST">
     <table border="0" width="750" cellspacing="1" cellpadding="1">
     <tr>
        <th colspan="2">ACTUALIZACIÓN DE DATOS DE PRODUCTO</th>
     </tr>
     <tr>
         <th>CODIGO</th>
         <td><input type="text" name="txtCodigo"
                    value="<?php echo $productos[0]; ?>" readonly /></td>
     </tr>
     <tr>
         <th>DESCRIPCION</th>
         <td><input type="text" name="txtDescripcion" size="80"
                    value="<?php echo $productos[1]; ?>" /></td>
```

```
            <td></td>
        </tr>
        <tr>
            <th>PRECIO</th>
            <td><input type="text" name="txtPrecio"  size="80"
                       value="<?php echo $productos[2]; ?>" /></td>
            <td></td>
        </tr>
        <tr>
            <th>STOCK</th>
            <td><input type="text" name="txtStock" size="80"
                       value="<?php echo $productos[3]; ?>" /></td>
            <td></td>
        </tr>
        <tr>
            <th>FECHA VENC.</th>
            <td><input type="text" name="txtFecha"  size="80"
                       value="<?php echo $productos[4]; ?>" /></td>
            <td></td>
        </tr>
        <tr>
            <td></td>
            <td><input type="submit" name="btnMantenimiento"
                       value="ACTUALIZA" /></td>
        </tr>
        </table>
    </form>
    </html>
```

Archivo: **elimina.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
    <link href="estilo.css" rel="stylesheet">
  </head>
  <body>
    <?php
      error_reporting(0);
      include('encabezado.php');
      if (isset($_POST['btnMantenimiento'])){
         $botón=$_POST['btnMantenimiento'];
         if ($botón=="AGREGAR NUEVO PRODUCTO"){
           include('nuevo.php');
           }elseif ($botón=="LISTADO DE PRODUCTOS"){
             include('listado.php');
           }elseif ($botón=="REGISTRAR"){
             include('producto. Php');
             include('productoCN.php');
             $objPro = new producto();
             $objNeg = new productoCN();

             $objPro->setCodigo($_POST['txtCodigo']);
             $objPro->setDescripcion($_POST['txtDescripcion']);
             $objPro->setPrecio($_POST['txtPrecio']);
             $objPro->setStock($_POST['txtStock']);
             $objPro->setFecha($_POST['txtFecha']);
```

```
            $objNeg->registra($objPro->getCodigo(),$objPro->getDescripcion(),
                              $objPro->getPrecio(),$objPro->getStock(),
                              $objPro->getFecha());

          }elseif ($boton=="ELIMINA"){
            include('producto.php');
            include('productoCN.php');
            $objPro = new producto();
            $objNeg = new productoCN();

            $objPro->setCodigo($_POST['txtCodigo']);

            $objNeg->elimina($objPro->getCodigo());
          }
        }
    ?>
    <?php
      include('productoCN.php');
      $objNeg = new productoCN();

      $código=$_GET['id'];

      $productos=$objNeg->buscaProducto($código);
    ?>

    <form method="POST">
    <table border="0" width="750" cellspacing="1" cellpadding="1">
    <tr>
       <th colspan="2">BAJA DE PRODUCTO</th>
    </tr>
    <tr>
        <td>CODIGO</td>
        <td><input type="text" name="txtCodigo"
                   value="<?php echo $productos[0]; ?>" readonly /></td>
    </tr>
    <tr>
        <td>DESCRIPCION</td>
        <td><?php echo $productos[1]; ?></td>
        <td></td>
    </tr>
    <tr>
        <td>PRECIO</td>
        <td><?php echo $productos[2]; ?></td>
        <td></td>
    </tr>
    <tr>
        <td>STOCK</td>
        <td><?php echo $productos[3]; ?></td>
        <td></td>
    </tr>
    <tr>
        <td>FECHA VENC.</td>
        <td><?php echo $productos[4]; ?></td>
        <td></td>
    </tr>
    <tr>
        <td></td>
        <td><input type="submit" name="btnMantenimiento"
                   value="ELIMINA" /></td>
    </tr>
    </table>
</form>
</html>
```

Archivo: **cerrar.php**

```
<?php
    mysql_close($cn);
?>
```

Archivo: **estilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
table {
    margin: auto;
}
#centrado{
    text-align: center;
}
#banner{
    margin: auto;
    display: block;
}
table th {
    background-color: #b9c9fe;
    text-align: left;
}
```

Caso desarrollado 5 **Mantenimiento de clientes con MVC**

Implementar una aplicación web con PHP que permita gestionar el mantenimiento de registros de la tabla Cliente usando el patrón MVC desde la base de datos VENTAS2015, tal como se muestra en la siguiente imagen:

Imagen: **index.php**

Tener en cuenta lo siguiente

- Implementar toda la aplicación en una carpeta llamada **MantenimientoClientesMVC**.
- Al iniciar la aplicación se deberá mostrar la página **index.php**.
- Del menú de opciones solo están habilitadas el menú Principal y Clientes, el primero muestra la página inicial llamada **index.php** y el segundo, opciones para el mantenimiento de registros de clientes usando MVC, las opciones que presenta son:

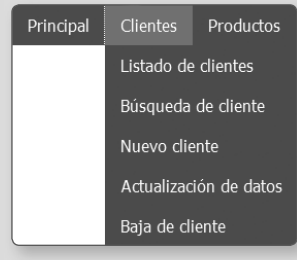


Imagen: Menú de opciones para el mantenimiento de clientes

- La primera opción del menú permite listar los clientes registrados hasta el momento, tal como se muestra en la siguiente imagen:

Principal Clientes Productos Distritos Boletas Venta Genera Reportes

LISTADO DE CLIENTES

CÓDIGO	CLIENTE	DIRECCIÓN	TELÉFONO	DISTRITO	EMAIL
C0001	CARLOS LOPEZ HURTADO	Av. Sangarara	4677352	BREÑA	clopez@hotmail.com
C0002	FRIDA QUIROZ DE LA CRUZ	Jr. Los Reyes 475	6734732	CARABAYLLO	fquiroz@peru.com
C0003	ALEJANDRO TRAUCO MANRIQUE	Jr.Huascar 956	4676732	BARRANCO	atruco@gmail.com
C0004	CARLA BLANCO RUIZ	Av. Arequipa 451	4867438	ATE	cblanco@hotmail.com
C0005	CORINA MENDOZA PEREZ	Av. Ingenieros 222	2548965	BREÑA	cmendoza@hotmail.com
C0006	JORGE RODAS DIONICIO	Jr. Las Liras 456	4789658	ATE	jrodas@gmail.com
C0007	ORLANDO CUEVAS CABANILLAS	Calle La Encantada 425	5698532	JESUS MARIA	ocuevas@peru.com
C0008	ANTUANE RODRIGUEZ ALARCON	Av. Dorado 347	4589732	BREÑA	arodriguez@hotmail.com
C0009	MANUEL SUAREZ FERNANDEZ	Jr. Los Robles 854	4576738	PACHACAMAC	msuarez@gmail.com
C0010	JUAN GUTIERREZ DIAZ	Calle Girasoles 456	4897421	LINCE	jgutierrez@hotmail.com
C0011	CARLOS COLAN BARDALES	Av. Los Heroes 895	3698574	PACHACAMAC	ccolan@peru.com
C0012	MARTIN CARRILLO SALAS	Calle Los Huertos 844	8965952	CHACLACAYO	mcarrillo@gmail.com
C0013	JOSE LAZARTE LUJAN	Jr. Agapito 452	1258965	MAGDALENA	jlazarte@gmail.com
C0014	VIDAL ZORRILLA RODRIGUEZ	Av. Heroes del cenepa 635	1547894	PUCUSANA	vzorrilla@hotmail.com
C0015	GUILLERMO RAMOS FLORES	Jr. Las Almendras 211	4587964	ATE	gramos@gmail.com

Todos los derechos reservados manueltorres@2015

Retornar a la página principal

Imagen: **listado.php**

- Para listar los datos de un determinado cliente, se emplea la opción **Búsqueda de cliente**, que a través del código de un cliente mostrará sus datos registrados, tal como se muestra en la siguiente imagen:

Imagen: **buscar.php**

Se debe tener en cuenta que los datos mostrados incluyen el nombre del distrito de donde proviene el cliente, de ninguna manera deberá mostrarse el código del distrito.

- Para registrar un nuevo cliente, se debe seleccionar la opción **Nuevo cliente**, que muestra el código autogenerado y a la vez precarga los distritos en el control cuadro combinado, tal como se muestra en la siguiente imagen:

Imagen: registrar.php

Tener en cuenta que para registrar un nuevo cliente, se deben ingresar los valores como nombres, apellidos, dirección, teléfono (solo 7 dígitos), seleccionar el distrito e ingresar el correo electrónico del nuevo cliente, para finalmente registrarlo en la base de datos con el botón **REGISTRAR**. Asimismo, el código autogenerado no puede ser modificado por el usuario.

- Para poder realizar modificaciones sobre los datos de un cliente, seleccionar la opción **Actualización de datos**, dicha ventana solicita un código de cliente; el cual buscará en la base de datos, una vez encontrado se pueden modificar sus valores, tal como se muestra en la siguiente imagen:

Imagen: actualizar.php

- Finalmente, se tiene la opción de dar de baja a un cliente, es decir, se puede eliminar un registro de cliente, previamente buscado, tal como se muestra en la siguiente imagen:

- Para las opciones de nuevo cliente, actualización de datos y baja de cliente, se debe tener en cuenta que una vez realizada su acción, se mostrará la página **listado.php**. Así, por ejemplo, si registramos a un nuevo cliente, al presionar el botón **REGISTRAR**, esta mostrará el listado de los clientes con los datos del nuevo cliente.
- La estructura final del proyecto es como sigue:

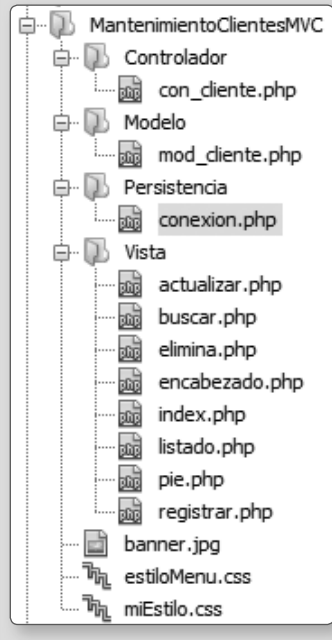


Imagen: Tomada del proyecto de Netbeans

Dentro de la carpeta principal del proyecto se debe crear cuatro subcarpetas los cuales tienen el siguiente objetivo:

- **Controlador**: Es el encargado de contener el archivo **con_cliente.php**, el cual contiene el método de registro del nuevo cliente.
- **Modelo**: Es el encargado de contener el archivo **mod_cliente**, el cual contiene la clase Cliente, método constructor y los métodos **Get/Set**.
- **Persistencia**: Es el encargado de contener el archivo **conexión.php**, el cual contiene las funciones referentes al mantenimiento de los registros del cliente, desde la conexión a la base de datos hasta la eliminación del registro de clientes.
- **Vista**: Es el encargado de contener todos los archivos que interactúan con el usuario desde la página principal (index.php) hasta la ventana de eliminación de registro (elimina.php).

El objetivo de cada archivo que contiene el proyecto es:

- **conexion.php**: Contiene los métodos necesarios para la conexión a la base de datos ventas2015, esto se definirá en el método constructor y, además, de los métodos que permiten realizar el mantenimiento de los registros como listado de cliente, listado de distritos, generación de código del nuevo cliente, registrar al nuevo cliente, buscar un determinado cliente, actualizar los datos de un cliente y eliminar el registro de cliente.
- **con_cliente.php**: Es la clase que contiene el método de registro del nuevo cliente.
- **mod_cliente.php**: Contiene la definición de la clase cliente, método constructor y los métodos **Get/Set**, esto nos permitirá usar los parámetros de registro, actualización, eliminación y buscado como objeto de la clase cliente.
- **encabezado.php**: Contiene la estructura y los enlaces que componen la parte superior de todas las páginas web contenidas en el proyecto y especialmente aquellas que interactúan con el usuario.
- **pie.php**: Contiene los créditos mostrados en todos los archivos del proyecto que interactúan con el usuario.
- **index.php**: Es la página principal, la cual inicia el proyecto de mantenimiento de registros.
- **listado.php**: Permite mostrar un listado de los clientes registrados en la base de datos ventas2015.
- **buscar.php**: Permite buscar los datos de un determinado cliente a través de su código.
- **registrar.php**: Permite registrar los datos de un nuevo cliente, previamente autogenerado el código.
- **actualizar.php**: Permite actualizar los datos de un determinado cliente, es decir, primero se debe buscar sus datos para luego actualizar algún dato mostrado.
- **elimina.php**: Permite eliminar un registro de cliente previamente buscado.
- **estiloMenu.css**: Contiene los estilos usados para el menú de opciones mostrado en el encabezado de página.
- **miEstilo.css**: Contiene los estilos usados en todos los archivos que contiene el proyecto de mantenimiento de clientes.

Archivo: **conexion.php**

```
<?php
    class conexion{
    private $cn=null;

    //Método constructor que permite conectarse al servidor
    //y selecciona la base de datos Ventas2015 al cual queremos conectarnos
    public function __construct() {
        $this->cn=mysql_connect('localhost','root','');
        mysql_select_db('Ventas2015',$this->cn);
    }

    //Creando un arreglo de los clientes
    public function listado(){
        $sql="SELECT ID_CLIENTE,CONCAT(NOMBRES,' ',PATERNO,' ',MATERNO),
                     DIRECCION,FONO,DESCRIPCION,EMAIL
              FROM CLIENTE C
              JOIN DISTRITO D ON C.ID_DISTRITO=D.ID_DISTRITO";
        $rs=mysql_query($sql,  $this->cn);
        while($misClientes=mysql_fetch_array($rs)){
            $clientes[]=$misClientes;
        }
        return $clientes;
    }

    //Creando un arreglo de los distritos
    public function listadoDistritos(){
        $sql="SELECT ID_DISTRITO,DESCRIPCION FROM DISTRITO";
        $rs=mysql_query($sql,  $this->cn);
        while($distritos=mysql_fetch_array($rs)){
            $distrito[]=$distritos;
        }
        return $distrito;
    }

    //Método que permite generar código nuevo de cliente
    public function generaCodigo(){
        $rs=mysql_query("SELECT ID_CLIENTE
                         FROM CLIENTE
                         ORDER BY ID_CLIENTE DESC LIMIT 1;",$this->cn);
        $fila = mysql_fetch_row($rs);
        return 'C'.str_pad((substr($fila[0], -4)+1),4,'0',STR_PAD_LEFT);;
    }

    //Método que permite registrar los datos de un nuevo cliente
    public function registra(mod_cliente $objCli){
        $rs=mysql_query("INSERT INTO CLIENTE(id_cliente,nombres,paterno,
                                             materno,direccion,fono,
                                             id_distrito,email)
                         VALUES ('{$objCli->getCodigo()}',
                                 '{$objCli->getNombres()}',
                                 '{$objCli->getPaterno()}',
                                 '{$objCli->getMaterno()}',
                                 '{$objCli->getDireccion()}',
                                 '{$objCli->getFono()}',
                                 '{$objCli->getDistrito()}',
                                 '{$objCli->getCorreo()}')",$this->cn);
    }

    //Método que permite buscar un determinado cliente por su código
    public function buscaCliente(mod_cliente $objCli){
```

```
        $rs=mysql_query("SELECT * FROM CLIENTE
                         WHERE ID_CLIENTE='{$objCli->getCodigo()}'",$this->cn);
            $cliente = mysql_fetch_array($rs);
            return $cliente;
        }

        //Método que permite actualizar un registro de cliente
        public function actualiza(mod_cliente $objCli){
          $rs=mysql_query("UPDATE CLIENTE
                              SET NOMBRES='{$objCli->getNombres()}',
                                  PATERNO='{$objCli->getPaterno()}',
                                  MATERNO='{$objCli->getMaterno()}',
                                  DIRECCION='{$objCli->getDireccion()}',
                                  FONO='{$objCli->getFono()}',
                                  ID_DISTRITO='{$objCli->getDistrito()}',
                                  EMAIL='{$objCli->getCorreo()}'
                            WHERE ID_CLIENTE='{$objCli->getCodigo()}'",$this->cn);
        }

        //Método que permite eliminar un determinado cliente basado en su código
        public function elimina(mod_cliente $objCli){
          $rs=mysql_query("DELETE FROM CLIENTE
                           WHERE ID_CLIENTE='{$objCli->getCodigo()}'",$this->cn);
        }
      }
    ?>
```

Archivo: **mod_cliente.php**

```
<?php
class mod_cliente {
    //Declaración de los atributos privados de la clase
    private $código;
    private $nombres;
    private $paterno;
    private $materno;
    private $direccion;
    private $fono;
    private $distrito;
    private $correo;

    //Implementación de los métodos GET
    public function getCodigo() {
        return $this->codigo;
    }

    public function getNombres() {
        return $this->nombres;
    }

    public function getPaterno() {
        return $this->paterno;
    }

    public function getMaterno() {
        return $this->materno;
    }

    public function getDireccion() {
        return $this->direccion;
```

```
    }

    public function getFono() {
        return $this->fono;
    }

    public function getDistrito() {
        return $this->distrito;
    }

    public function getCorreo() {
        return $this->correo;
    }

    //Implementación de los métodos SET
    public function setCodigo($código) {
        $this->código = $código;
    }

    public function setNombres($nombres) {
        $this->nombres = $nombres;
    }

    public function setPaterno($paterno) {
        $this->paterno = $paterno;
    }

    public function setMaterno($materno) {
        $this->materno = $materno;
    }

    public function setDireccion($direccion) {
        $this->direccion = $direccion;
    }

    public function setFono($fono) {
        $this->fono = $fono;
    }

    public function setDistrito($distrito) {
        $this->distrito = $distrito;
    }

    public function setCorreo($correo) {
        $this->correo = $correo;
    }

    //Implementación del método constructor
    function __construct($código, $nombres, $paterno, $materno,
                         $direccion, $fono, $distrito, $correo) {
        $this->código = $código;
        $this->nombres = $nombres;
        $this->paterno = $paterno;
        $this->materno = $materno;
        $this->direccion = $direccion;
        $this->fono = $fono;
        $this->distrito = $distrito;
        $this->correo = $correo;
    }
}
```

Archivo: **con_cliente.php**

```
<?php
    //Incluimos los métodos públicos dentro del archivo con_cliente.php
    include '../Modelo/mod_cliente.php';
    include '../Persistencia/conexion.php';

    //Creamos un objeto de la clase conexión para poder acceder a los métodos
    //de mantenimiento de los clientes.
    $conexion = new conexion();

    //Validamos la selección del botón Registrar para poder capturar todos
    //los valores ingresados para el nuevo cliente.
    if (isset($_POST['btnRegistrar'])){
        $codigo=$_POST['txtCodigo'];
        $nombres=$_POST['txtNombres'];
        $paterno=$_POST['txtPaterno'];
        $materno=$_POST['txtMaterno'];
        $direccion=$_POST['txtDireccion'];
        $telefono=$_POST['txtTelefono'];
        $distrito=$_POST['selDistrito'];
        $email=$_POST['txtEmail'];

        //Creamos un objeto de la clase modCliente enviando los valores
        //capturados al constructor de la clase.
        $modCliente=new mod_cliente($código, $nombres, $paterno,
                                    $materno, $direccion, $telefono,
                                    $distrito, $email);

        //Ejecutamos el registro mediante la invocación del método registra
        //que se encuentra en la clase conexión.
        $conexion->registra($modCliente);
        header('location:../Vista/listado.php');
    }
?>
```

Archivo: **encabezado.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
    <link href="../estiloMenu.css" rel="stylesheet">
  </head>
  <body>
    <img src="../banner.jpg" width="700" height="200" />
    <header>
      <ul class="nav">
        <li><a href="index.php">Principal</a></li>
        <li><a href="#">Clientes</a>
          <ul>
            <li><a href="listado.php">Listado de clientes</a></li>
            <li><a href="buscar.php">Búsqueda de cliente</a></li>
            <li><a href="registrar.php">Nuevo cliente</a></li>
            <li><a href="actualizar.php">Actualización de datos</a></li>
            <li><a href="elimina.php">Baja de cliente</a></li>
          </ul>
        </li>
        <li><a href="#">Productos</a>
```

```
        <ul>
              <li><a href="#">Listado de productos</a></li>
              <li><a href="#">Nuevo producto</a></li>
              <li><a href="#">Actualización de datos</a></li>
              <li><a href="#">Baja de producto</a></li>
            </ul>
          </li>
          <li><a href="#">Distritos</a>
             <ul>
               <li><a href="#">Listado de distritos</a></li>
               <li><a href="#">Nuevo distrito</a></li>
               <li><a href="#">Actualización de datos</a></li>
             </ul>
           </li>
           <li><a href="#">Boletas</a>
             <ul>
                <li><a href="#">Listado de boletas</a></li>
                <li><a href="#">Listado de boletas por cliente</a></li>
             </ul>
           </li>
           <li><a href="#">Venta</a>
             <ul>
               <li><a href="#">Nueva Venta</a></li>
               <li><a href="#">Consulta</a></li>
               <li><a href="#">Listado</a></li>
             </ul>
           </li>
           <li><a href="#">Genera Reportes</a>
             <ul>
                <li><a href="#">Clientes</a></li>
                <li><a href="#">Productos</a></li>
                <li><a href="#">Distritos</a></li>
                <li><a href="#">Boletas</a></li>
                <li><a href="#">Detalle por boleta</a></li>
              </ul>
            </li>
          </ul>
    </header>
    <br><br><br>
  </body>
</html>
```

Archivo: **pie.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
    <link href="../miEstilo.css" rel="stylesheet">
  </head>
  <body>
    <h5 id="centrado">Todos los derechos reservados manueltorres@2015</
h5>
    <p id="centrado">
           <a href="index.php">Retornar a la página principal</a>
    </p>
  </body>
</html>
```

Archivo: **index.php**

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>SISTEMA DE CONTROL DE VENTAS</title>
        <link href="../miEstilo.css" rel="stylesheet" >
    </head>
    <body>
        <header>
            <?php
                include('encabezado.php');
            ?>
        </header>
        <footer>
            <?php include('pie.php'); ?>
        </footer>
    </body>
</html>
```

Archivo: **listado.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>SISTEMA DE CONTROL DE VENTAS</title>
    <link href="../miEstilo.css" rel="stylesheet" >
  </head>
  <body>
    <header>
      <?php
         include('encabezado.php');
      ?>
    </header>
    <section>
      <h3>LISTADO DE CLIENTES</h3>
      <?php
         //Agregamos los métodos implementados en la clase conexión.php
         include('../Persistencia/conexion.php');

         //Creamos un objeto de la clase conexión para poder acceder a los
         //métodos.
         $objCon = new conexion();

         //Invocamos al método que lista los clientes mediante el objeto
         //asociado a la clase conexión.
         $cliente=$objCon->listado();
      ?>
      <table border="1" width="700" cellspacing="1" cellpadding="1">
        <tr>
          <td>CODIGO</td>
          <td>CLIENTE</td>
          <td>DIRECCION</td>
          <td>TELEFONO</td>
          <td>DISTRITO</td>
          <td>EMAIL</td>
        </tr>
```

```
        <?php
          //Ahora recorremos por todos los registros para poder colocarlos
          //en una tabla para visualizarlo.
          foreach ($cliente as $cli){
        ?>
        <tr>
           <td><?php echo $cli[0]; ?></td>
           <td><?php echo $cli[1]; ?></td>
           <td><?php echo $cli[2]; ?></td>
           <td><?php echo $cli[3]; ?></td>
           <td><?php echo utf8_decode($cli[4]); ?></td>
           <td><?php echo $cli[5]; ?></td>
        </tr>
        <?php } ?>
      </table>
    </section>
    <footer>
      <?php include('pie.php'); ?>
    </footer>
  </body>
</html>
```

Comentarios

```
<tr>
   <td><?php echo $cli[0]; ?></td>
   <td><?php echo $cli[1]; ?></td>
   <td><?php echo $cli[2]; ?></td>
   <td><?php echo $cli[3]; ?></td>
   <td><?php echo utf8_decode($cli[4]); ?></td>
   <td><?php echo $cli[5]; ?></td>
</tr>
```

La estructura **foreach** llena la variable **$cli** con los datos de los clientes, los cuales pueden ser invocados de dos formas, la primera se puede realizar mediante el índice del resultado, es decir, 0 el código, 1 los datos completos del cliente, 2 para la dirección, 3 para el teléfono, 4 para el nombre del distrito y 5 para el correo electrónico del cliente. La otra forma sería especificar el nombre del campo como $cli['ID_CLIENTE'] para el código y así, sucesivamente, para los demás campos.

Archivo: **buscar.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>SISTEMA DE CONTROL DE VENTAS</title>
    <link href="../miEstilo.css" rel="stylesheet" >
  </head>
  <body>
    <header>
      <?php
         //Ocultamos los errores de advertencia de PHP
         error_reporting(0);
         include('encabezado.php');
```

```
        //Agregamos todos los métodos definidos en la clase con_cliente
        include '../Controlador/con_cliente.php';

        //Creamos un objeto de la clase conexión
        $objCon = new conexion();
      ?>
    </header>
    <section>
      <?php
        //Creamos un objeto de la clase mod_cliente
        $modCliente = new mod_cliente();

      //Enviamos el código del cliente a la clase ya que el método de
      //búsqueda necesita un objeto y no un valor.
        $modCliente->setCodigo($_POST['txtCodigo']);

      //Invocamos al método buscaCliente enviándole el objeto modCliente
      //el cual contiene solo el código del cliente
        $cliente=$objCon->buscaCliente($modCliente);
      ?>
      <h3>BUSQUEDA DE DATOS DE CLIENTE</h3>
      <form method="POST">
        <table>
          <tr>
            <td>CODIGO</td>
            <td><input type="text" name="txtCodigo"
                       value="<?php echo $_POST['txtCodigo']; ?>"/>
                <input type="submit" name="btnBuscar" value="BUSCAR" />
            </td>
          </tr>
          <tr>
            <td>NOMBRES</td>
            <td><?php echo $cliente[1]; ?></td>
          </tr>
          <tr>
            <td>APELLIDO PATERNO</td>
            <td><?php echo $cliente[2]; ?></td>
          </tr>
          <tr>
            <td>APELLIDO MATERNO</td>
            <td><?php echo $cliente[3]; ?></td>
          </tr>
          <tr>
            <td>DIRECCION</td>
            <td><?php echo $cliente[4]; ?></td>
          </tr>
          <tr>
            <td>TELEFONO</td>
            <td><?php echo $cliente[5]; ?></td>
          </tr>
          <tr>
            <td>DISTRITO</td>
            <td>
              <?php
                //Enviando el resultado del listado de los distritos a la
                 //variable $distrito.
                 $distrito=$objCon->listadoDistritos();

                 //Recorremos por todos los distritos para comprobar el código
                 //del distrito del cliente con el código de la propia tabla
                 //distrito y así poder retornar el nombre del distrito.
```

```
                    //Se asignó break para que una vez que lo encuentre ya no
                    //continúe buscando.
                    foreach ($distrito as $dist){
                       if ($cliente[6] == $dist[0]){
                          echo $dist[1];
                          break;
                       }
                    }
                 ?>
                 </td>
              </tr>
              <tr>
                <td>EMAIL</td>
                <td><?php echo $cliente[7]; ?></td>
              </tr>
          </table>
        </form>
    </section>
    <footer>
      <?php include('pie.php'); ?>
    </footer>
  </body>
</html>
```

Comentarios

```
<tr>
  <td>NOMBRES</td>
  <td><?php echo $cliente[1]; ?></td>
</tr>
```

En la tabla se mostrará los datos del cliente encontrados a partir de la variable **$cliente**, que recibió el registro desde la invocación al método buscaCliente, considere que 1 representa a la segunda columna del resultado. Asimismo, se aplicarán a las demás celdas con los datos faltantes del cliente.

Archivo: **registrar.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>SISTEMA DE CONTROL DE VENTAS</title>
    <link href="../miEstilo.css" rel="stylesheet" >
  </head>
  <body>
    <header>
      <?php
        include('encabezado.php');
        include '../Persistencia/conexion.php';
        $objCon = new conexion();
      ?>
    </header>
    <section>
      <h3>REGISTRO DEL NUEVO PRODUCTO</h3>
      <form action="../Controlador/con_cliente.php" method="POST">
```

```
          <table>
            <tr>
              <td>CODIGO</td>
              <td><input type="text" name="txtCodigo"
                     value="<?php echo $objCon->generaCodigo(); ?>" readonly/>
              </td>
            </tr>
            <tr>
               <td>NOMBRES</td>
               <td><input type="text" name="txtNombres" value="" /></td>
            </tr>
            <tr>
               <td>APELLIDO PATERNO</td>
               <td><input type="text" name="txtPaterno" value="" /></td>
            </tr>
            <tr>
               <td>APELLIDO MATERNO</td>
               <td><input type="text" name="txtMaterno" value="" /></td>
            </tr>
            <tr>
               <td>DIRECCION</td>
               <td><input type="text" name="txtDireccion" value="" /></td>
            </tr>
            <tr>
               <td>TELEFONO</td>
               <td><input type="text" name="txtTelefono" value="" /></td>
            </tr>
            <tr>
               <td>DISTRITO</td>
               <td><select name="selDistrito">
                     <?php
                              //El resultado de listar los distritos es enviado a la
                              //variable $distrito.
                        $distrito=$objCon->listadoDistritos();

                        //Recorremos por todos los distritos con el objetivo
                        //de llenar el cuadro combinado
                        foreach ($distrito as $dist){
                           echo "<option value='".$dist[0]."'>"
                                                  .$dist[1]."</option>";
                        }
                     ?>
                  </select>
               </td>
            </tr>
            <tr>
              <td>EMAIL</td>
              <td><input type="text" name="txtEmail" value="" /></td>
            </tr>
            <tr>
              <td><input type="submit" name="btnRegistrar"
                         value="REGISTRAR" /></td>
              <td></td>
            </tr>
          </table>
        </form>
      </section>
      <footer>
         <?php include('pie.php'); ?>
      </footer>
    </body>
  </html>
```

Comentarios

```
<form action="../Controlador/con_cliente.php" method="POST">
```

La característica del archivo **registrar.php** es que el archivo no registra los datos del cliente, esta labor lo realiza la clase **con_cliente**, es por ese motivo que al momento de implementar el formulario se debe considerar la propiedad action con el acceso a la clase **con_cliente** y en el método especificar **POST**.

Archivo: **actualizar.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>SISTEMA DE CONTROL DE VENTAS</title>
    <link href="../miEstilo.css" rel="stylesheet" >
  </head>
  <body>
    <header>
       <?php
          //Omitimos los errores de advertencia de PHP
          error_reporting(0);
          include('encabezado.php');

          //Incluimos los métodos que permiten conectarse a la base de datos
          include '../Persistencia/conexion.php';

          //Incluimos los métodos GET y SET de la clase cliente
          include '../Modelo/mod_cliente.php';

          //Creamos un objeto de la conexión para tener acceso a los métodos
          //de búsqueda y actualización de datos del cliente.
          $objCon = new conexion();

          //Creamos el objeto $modCliente para tener acceso a los
          //métodos GET y SET de la clase cliente.
          $modCliente = new mod_cliente();
          $modCliente->setCodigo($_POST['txtCodigo']);
       ?>
    </header>
    <section>
       <?php
          //Determina qué botón ha seleccionado el usuario
          if (isset($_POST['btnGenerico'])) {

             //Capturar el botón seleccionado
             $botón = $_POST['btnGenerico'];

             //Si el botón es Buscar se ejecutará el método buscaCliente
             if ($botón == 'BUSCAR') {
               $cliente=$objCon->buscaCliente($modCliente);
             }

             //Si el botón es Actualiza entonces capturar todos los valores y
             //los envía al método constructor de la clase mod_cliente para
```

```
        //actualizarlo con el método actualiza. Finalmente se invoca al
        //archivo listado.php que permite listar los cliente con los
        //cambios realizados
        if ($botón == 'ACTUALIZA'){
          $código=$_POST['txtCodigo'];
          $nombres=$_POST['txtNombres'];
          $paterno=$_POST['txtPaterno'];
          $materno=$_POST['txtMaterno'];
          $direccion=$_POST['txtDireccion'];
          $telefono=$_POST['txtTelefono'];
          $distrito=$_POST['selDistrito'];
          $email=$_POST['txtEmail'];
          $modCliente=new mod_cliente($código, $nombres,
                                      $paterno, $materno,
                                      $direccion, $telefono,
                                      $distrito, $email);
          $objCon->actualiza($modCliente);
          header('location:../Vista/listado.php');
        }
    }
?>
<h3>ACTUALIZACIÓN DE DATOS DE CLIENTES</h3>
<form method="POST">
  <table>
    <tr>
      <td>CODIGO</td>
      <td><input type="text" name="txtCodigo"
                 value="<?php echo $_POST['txtCodigo']; ?>"/>
          <input type="submit" name="btnGenerico" value="BUSCAR" />
      </td>
    </tr>
    <tr>
       <td>NOMBRES</td>
       <td><input type="text" name="txtNombres"
                  value="<?php echo $cliente[1]; ?>" />
       </td>
    </tr>
    <tr>
      <td>APELLIDO PATERNO</td>
      <td><input type="text" name="txtPaterno"
                 value="<?php echo $cliente[2]; ?>" />
      </td>
    </tr>
    <tr>
       <td>APELLIDO MATERNO</td>
       <td><input type="text" name="txtMaterno"
                  value="<?php echo $cliente[3]; ?>" />
        </td>
    </tr>
    <tr>
       <td>DIRECCION</td>
       <td><input type="text" name="txtDireccion"
                  value="<?php echo $cliente[4]; ?>" />
       </td>
    </tr>
    <tr>
       <td>TELEFONO</td>
       <td><input type="text" name="txtTelefono"
                  value="<?php echo $cliente[5]; ?>" /></td>
    </tr>
    <tr>
       <td>DISTRITO</td>
```

```
            <td>
               <select name="selDistrito">
                 <?php
                      $distrito=$objCon->listadoDistritos();
                      foreach ($distrito as $dist){
                          if ($cliente[6] == $dist[0])
                           $seleccionado='SELECTED';
                           else
                           $seleccionado='';
                 ?>
                 <option VALUE="<?php echo $dist[0]; ?>"
                          <?php echo $seleccionado; ?>>
                          <?php echo $dist[1] ?>
                 </option>
                 <?php } ?>
               </select>
             </td>
          </tr>
          <tr>
             <td>EMAIL</td>
             <td><input type="text" name="txtEmail"
                        value="<?php echo $cliente[7]; ?>" /></td>
          </tr>
          <tr>
            <td><input type="submit" name="btnGenerico"
                       value="ACTUALIZA" /></td>
            <td></td>
          </tr>
       </table>
    </form>
  </section>
  <footer>
     <?php include('pie.php'); ?>
  </footer>
 </body>
</html>
```

Comentarios

```
<tr>
  <td>DISTRITO</td>
  <td>
    <select name="selDistrito">
    <?php
      $distrito=$objCon->listadoDistritos();
      foreach ($distrito as $dist){
        if ($cliente[6] == $dist[0])
        $seleccionado='SELECTED';
        else
        $seleccionado='';
    ?>
      <option VALUE="<?php echo $dist[0]; ?>"
                    <?php echo $seleccionado; ?>>
                    <?php echo $dist[1] ?>
      </option>
    <?php } ?>
   </select>
  </td>
</tr>
```

El objetivo del script es mostrar el nombre del distrito buscado, se debe recordar que al momento de buscar un cliente, este no cuenta con el nombre del distrito sino más bien con el código del mismo, es allí donde se debe hacer una comparación con el código del distrito almacenado en la tabla Distrito, si ambos códigos coinciden que muestren el nombre del distrito encontrado, asignando SELECTED al código del distrito encontrado en la búsqueda. Recuerde que $cliente[6] contiene el código del distrito desde la tabla Cliente, $dist[0] también contiene el código del distrito pero de la tabla Distrito y $dist[1] tiene la descripción del distrito.

Archivo: **elimina.php**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>SISTEMA DE CONTROL DE VENTAS</title>
    <link href="../miEstilo.css" rel="stylesheet" >
  </head>
  <body>
    <header>
      <?php
        error_reporting(0);
        include('encabezado.php');
        include '../Modelo/mod_cliente.php';
        include '../Persistencia/conexion.php';
        $objCon = new conexion();
      ?>
    </header>
    <section>
      <?php
        $modCliente = new mod_cliente();
        $modCliente->setCodigo($_POST['txtCodigo']);

        if (isset($_POST['btnGenerico'])) {
          $boton = $_POST['btnGenerico'];
          if ($boton == 'BUSCAR') {
            $cliente=$objCon->buscaCliente($modCliente);
          }
          if ($botón == 'ELIMINA'){
            $objCon->elimina($modCliente);
            header('location:../Vista/listado.php');
          }
        }
      ?>
      <h3>BAJA DE REGISTRO DE CLIENTE</h3>
      <form method="POST">
        <table>
          <tr>
            <td>CODIGO</td>
            <td><input type="text" name="txtCodigo"
                       value="<?php echo $_POST['txtCodigo']; ?>"/>
                <input type="submit" name="btnGenerico" value="BUSCAR" />
            </td>
          </tr>
          <tr>
            <td>NOMBRES</td>
            <td><?php echo $cliente[1]; ?></td>
          </tr>
          <tr>
```

```
            <td>APELLIDO PATERNO</td>
            <td><?php echo $cliente[2]; ?></td>
          </tr>
          <tr>
            <td>APELLIDO MATERNO</td>
            <td><?php echo $cliente[3]; ?></td>
          </tr>
          <tr>
            <td>DIRECCION</td>
            <td><?php echo $cliente[4]; ?></td>
          </tr>
          <tr>
            <td>TELEFONO</td>
            <td><?php echo $cliente[5]; ?></td>
          </tr>
          <tr>
            <td>DISTRITO</td>
            <td><?php echo $cliente[6] ?></td>
          </tr>
          <tr>
            <td>EMAIL</td>
            <td><?php echo $cliente[7]; ?></td>
          </tr>
          <tr>
             <td><input type="submit" name="btnGenerico"
                        value="ELIMINA" /></td>
             <td></td>
          </tr>
        </table>
      </form>
    </section>
    <footer>
      <?php include('pie.php'); ?>
    </footer>
  </body>
</html>
```

Archivo: **estiloMenu.css**

```
* {
 padding:0px;
 margin:0px;
}

header {
 margin:auto;
 width:700px;
 font-family:tahoma;
}

ul, ol {
 list-style:none;
}

.nav li a {
 background-color:#069;
 color:#FFFFFF;
 text-decoration:none;
 padding: 10px 15px;
```

```
 display:block;
}

.nav li a:hover {
 background-color:#FF3300;
}

.nav > li {
 float:left;
}

.nav li ul {
 display:none;
 position:absolute;
 min-width:100px;
}

.nav li:hover > ul {
 display:block;
}

.nav li ul li {
 position:relative;
}

.nav li ul li ul {
 right:-140px;
 top:0px;
}
```

Archivo: **miEstilo.css**

```
body {
    font-family: tahoma;
}

h3,h5{
    text-align: center;
}

img{
    margin: auto;
    display: block;
}

#centrado{
    text-align: center;
}

table {
    font-family: tahoma;
    font-size: 12px;
    margin: auto;
    width: 700px;
    text-align: left;
    border-collapse: collapse;
}

th {
```

```
    font-size: 13px;
    font-weight: normal;
    padding: 3px;
    background: #b9c9fe;
    border-top: 2px solid #aabcfe;
    border-bottom: 1px solid #fff; color: #039;
}

td {
    padding: 6px;
    background: #e8edff;
    border: 1px solid #fff;
    color: #669;
}

tr:hover td {
    background: #d0dafd;
    color: #339;
}

ul.dropdown-vertical {
 width: 200px;
}

ul.dropdown-vertical ul {
 top: 1px;
 left: 99%;
}

ul.dropdown-vertical li {
 float: none;
}
```

CAP.

7

Objetos de datos PHP (PDO)

7.1 Introducción

PDO es la inicial de «Objeto de datos en PHP», el cual define una nueva forma de conexión al servidor de base de datos. Los métodos de clase PDO permiten seleccionar una base de datos e implementar funciones que agilizan los resultados promoviendo la abstracción sobre el acceso a datos.

A lo largo de su uso, notará que un método implementado podrá ser reutilizado las veces que crea necesario, seguro que ahora se preguntará, si eso se hizo con la programación orientada a objetos en PHP, *¿cuál es la diferencia de usar la clase PDO?* La respuesta podría resumirse en que una sola definición de PDO permite ejecutar una sentencia SQL que no necesariamente tiene los mismos resultados. Dentro de ese marco, se puede decir que si implementamos una función de consulta sin parámetros, esa función podrá realizarla con uno o más parámetros sin modificar la estructura inicial del método; a diferencia de solo usar programación orientada al objeto en la cual tendríamos que crear un método para cada tipo de consulta.

Dentro de este marco, podemos decir que PDO suministra al programador una capa de abstracción de acceso a datos, es decir, no importa la base de datos que se esté usando; sino más bien, se usan las mismas funciones para realizar cualquier tipo de consultas y obtener datos de la mejor manera posible.

Finalmente, PDO viene incorporado a partir de la versión PHP 5.1 en adelante, además requiere características nuevas de la programación orientada a objetos 1, es así que una aplicación realizada en una versión no podrá ejecutarla en versiones anteriores.

7.2 Conexión PDO

A. Definición

Una conexión PDO se establece creando una instancia a la clase PDO, se debe tener en cuenta que PDO no se rige al controlador de base de datos de su aplicación, ya que siempre antecede la palabra PDO sobre sus métodos.

Por otra parte, posee un método constructor que solicita el origen de la data, el nombre del usuario y la clave del motor de base de datos que use en su aplicación, en esta parte tiene un gran parecido a la función **mysql_connect**.

B. Sintaxis a MySQL

```
$cn=new PDO('mysql:host=localhost;dbname=base_datos','usuario','clave');
```

- mysql:host=localhost;dbname=base_datos: Aquí se determina el nombre del servidor localhost y dbname especifica el nombre de la base de datos.
- Usuario: Es el nombre del usuario configurado en el servidor de base de datos.
- Clave: Es la clave asignada al usuario si las tuviera.

C. Sintaxis usando control de errores

Se verá la sintaxis para la conexión con MySQL, usando control de errores, esto se puede ocasionar por una mala conexión al servidor o a la base de datos, para lo cual PDO lanza una excepción por medio de la clase PDOException.

```
try {
$cn=new PDO('mysql:host=localhost;dbname=base_datos','usuario','clave');
}catch (PDOException $ex) {
 echo "Hubo un error  " . $ex->getMessage();
}
```

D. Función de conexión PDO

En el capítulo 6 se dedicó el tema de Programación orientada a objetos, que puede ser embebida con la clase PDO, de forma que se pueden implementar funciones de conexión, la cual presenta la siguiente estructura:

```
$cn=null;
function getConecta(){
    try{
       $cadena='mysql:host=localhost;dbname=ventas2015';
       $this->cn = new PDO($cadena,'root','');
       $this->cn->setAttribute(PDO::ATTR_CASE,PDO::CASE_LOWER);
       return $this->cn;

    } catch(PDOException $ex){
       echo 'Error en la conexion - '.$ex->getMessage();
    }
}
```

- La variable $cadena contiene el script de conexión, tanto para el servidor como para la base de datos.
- $this->cn es el objeto asociado a la clase PDO, es aquí donde el constructor de dicha clase solicita datos, como el script de conexión, el usuario y la clave.
- $this->cn->setAttribute(PDO::ATTR_CASE,PDO::CASE_LOWER) es la sentencia mínima establecida para una conexion exitosa con la clase PDO. Donde PDO::ATTR_CASE fuerza a los nombres de las columnas de una tabla el uso de mayúsculas/minúsculas; mientras que PDO::CASE_LOWER fuerza a los nombres de las columnas de una tabla a expresarse en minúsculas.
- Finalmente, se retorna la conexión mediante la sentencia return $this->cn.

7.3 Casos desarrollados

Caso desarrollado 1 **Listado de clientes**

Implementar una aplicación web con PHP que permita listar los clientes haciendo uso de la clase PDO, tal como se muestra en la siguiente imagen:

LISTADO DE CLIENTES - USANDO LA CLASE PDO

CÓDIGO	CLIENTE	DIRECCIÓN	TELÉFONO	DISTRITO	EMAIL
C0001	CARLOS LOPEZ HURTADO	Av. Sangarara	4677352	BREÑA	clopez@hotmail.com
C0002	FRIDA QUIROZ DE LA CRUZ	Jr. Los Reyes 475	6734732	CARABAYLLO	fquiroz@peru.com
C0003	ALEJANDRO TRAUCO MANRIQUE	Jr.Huascar 956	4676732	BARRANCO	atruco@gmail.com
C0004	CARLA BLANCO RUIZ	Av. Arequipa 451	4867438	ATE	cblanco@hotmail.com
C0005	CORINA MENDOZA PEREZ	Av. Ingenieros 222	2548965	BREÑA	cmendoza@hotmail.com
C0006	JORGE RODAS DIONICIO	Jr. Las Liras 456	4789658	ATE	jrodas@gmail.com
C0007	ORLANDO CUEVAS CABANILLAS	Calle La Encantada 425	5698532	JESUS MARIA	ocuevas@peru.com
C0008	ANTUANE RODRIGUEZ ALARCON	Av. Dorado 347	4589732	BREÑA	arodriguez@hotmail.com
C0009	MANUEL SUAREZ FERNANDEZ	Jr. Los Robles 854	4576738	PACHACAMAC	msuarez@gmail.com
C0010	JUAN GUTIERREZ DIAZ	Calle Girasoles 456	4897421	LINCE	jgutierrez@hotmail.com
C0011	CARLOS COLAN BARDALES	Av. Los Heroes 895	3698574	PACHACAMAC	ccolan@peru.com
C0012	MARTIN CARRILLO SALAS	Calle Los Huertos 844	8965952	CHACLACAYO	mcarrillo@gmail.com
C0013	JOSE LAZARTE LUJAN	Jr. Agapito 452	1258965	MAGDALENA	jlazarte@gmail.com
C0014	VIDAL ZORRILLA RODRIGUEZ	Av. Heroes del cenepa 635	1547894	PUCUSANA	vzorrilla@hotmail.com
C0015	GUILLERMO RAMOS FLORES	Jr. Las Almendras 211	4587964	ATE	gramos@gmail.com

Todos los derechos reservados - manuelTorresR@2015

Imagen: listado.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **consultasPDO**.
- Al iniciar la aplicación deberá mostrar la página **listado.php**, la cual mostrará los datos de los clientes como código, nombre completo del cliente, dirección, teléfono, nombre del distrito y correo electrónico.
- Implementar un procedimiento que permita listar los datos de los clientes.

El objetivo de cada archivo que contiene el proyecto es:

- **Conexion.php**: Es la clase que contiene la función **getConecta** la cual permite conectarse al servidor y seleccionar la base de datos usando la clase PDO.
- **listado.php**: Contiene la página web que interactúa con el usuario y que permite mostrar los datos de los clientes.
- **miEstilo.css**: Contiene los estilos usados en el archivo **listado.php**.

Procedimiento almacenado

```
DROP PROCEDURE IF EXISTS SP_LISTADOCLIENTES;
CREATE PROCEDURE SP_LISTADOCLIENTES()
    SELECT C.ID_CLIENTE,
            CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
            C.DIRECCION,C.FONO,D.DESCRIPCION,C.EMAIL
            FROM CLIENTE C
            INNER JOIN DISTRITO D
            ON C.ID_DISTRITO=D.ID_DISTRITO
            ORDER BY 1 ASC;
```

Archivo: **Conexion.php**

```
<?php
class Conexion {
    //Atributo para la cadena de conexión
    private $cn=null;

    //Función de conexión
    public function getConexion(){
    try{
      //Script de la cadena de conexión
      $cadena='mysql:host=localhost;dbname=ventas2015';

      //Implementando la cadena de conexión
     $this->cn = new PDO($cadena,'root','');
      $this->cn->setAttribute(PDO::ATTR_CASE,PDO::CASE_LOWER);

      //Retornando la conexión correcta
      return $this->cn;
     } catch(PDOException $ex){
            echo 'Error en la conexion - '.$ex->getMessage();
      }
    }
}
```

Archivo: **listado.php**

```
<?php include('Conexion.php'); ?>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
        <link href="miEstilo.css" rel="stylesheet">
    </head>
    <body>
      <header>
          <?php include 'encabezado.php'; ?>
          <h3>LISTADO DE CLIENTES - USANDO LA CLASE PDO</h3>
      </header>
      <section>
         <?php
            //Creando el objeto de conexión
            $objCon = new Conexion();

            //Script de la consulta
            $consulta='CALL SP_LISTADOCLIENTES();';

            //Implementando la consulta y enviando el resultado a $rs
            $rs=$objCon->getConexion()->query($consulta);
        ?>
        <table border="0" width="800" cellspacing='0' cellpadding="0">
            <tr>
                <th>CODIGO</th>
                <th>CLIENTE</th>
                <th>DIRECCION</th>
                <th>TELEFONO</th>
                <th>DISTRITO</th>
                <th>EMAIL</th>
            </tr>
        <?php
            //Recorriendo por los registros consultados
```

```
            while($fila=$rs->fetch()){
        ?>
            <tr>
                <td><?php echo $fila[0]; ?></td>
                <td><?php echo $fila[1]; ?></td>
                <td><?php echo $fila[2]; ?></td>
                <td><?php echo $fila[3]; ?></td>
                <td><?php echo utf8_decode($fila[4]); ?></td>
                <td><?php echo $fila[5]; ?></td>
            </tr>
        <?php } ?>
        </table>
    </section>
    <footer>
        <h5>Todos los derechos reservados - manuelTorresR@2015</h5>
    </footer>
  </body>
</html>
```

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
table {
    margin: auto;
}
img{
    margin: auto;
    display: block;
}
td {
    border: solid 1px #006699;
    border-top: #006699;
    border-right: #006699;
    border-left: #006699;
    border-bottom: #006699;
}
table th {
background-color: #b9c9fe;
}
#centrado{
    text-align: center;
}
```

Caso desarrollado 2 **Consulta de datos de cliente por código**

Implementar una aplicación web con PHP que permita realizar la venta de productos, tal como se muestra en la siguiente imagen:

Imagen: consultaClientexCodigo.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **consultasPDO**.
- Al iniciar la aplicación deberá mostrar la página **consultaClientexCodigo.php**.
- Al ingresar un código válido podrá seleccionar el botón **Consultar** para mostrar los datos encontrados según un determinado cliente.
- Implementar un procedimiento almacenado que permita consultar los datos de un determinado cliente según su código.

El objetivo de cada archivo que contiene el proyecto es:

- **Conexion.php**: Es la clase que contiene la función **getConecta**, la cual permite conectarse al servidor y seleccionar la base de datos usando la clase PDO.
- **consultaClientexCodigo.php**: Contiene la página web que interactúa con el usuario y que permite mostrar los datos de los clientes según el código buscado.
- **miEstilo.css**: Contiene los estilos usados en el archivo **consultaClientexCodigo.php**.

Procedimiento almacenado

```
DROP PROCEDURE IF EXISTS SP_LISTADOCLIENTESXCODIGO;
CREATE PROCEDURE SP_LISTADOCLIENTESXCODIGO(IDC CHAR(5))
    SELECT C.ID_CLIENTE,
           CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
           C.DIRECCION,C.FONO,D.DESCRIPCION,C.EMAIL
           FROM CLIENTE C
           INNER JOIN DISTRITO D
           ON C.ID_DISTRITO=D.ID_DISTRITO
           WHERE C.ID_CLIENTE=IDC;
```

Archivo: **Conexion.php**

```
<?php
class Conexion {
    //Atributo para la cadena de conexión
    private $cn=null;
    //Función de conexión
    public function getConexion(){
    try{
      //Script de la cadena de conexión
      $cadena='mysql:host=localhost;dbname=ventas2015';

      //Implementando la cadena de conexión
    $this->cn = new PDO($cadena,'root','');
      $this->cn->setAttribute(PDO::ATTR_CASE,PDO::CASE_LOWER);

      //Retornando la conexión correcta
      return $this->cn;
     } catch(PDOException $ex){
           echo 'Error en la conexion - '.$ex->getMessage();
      }
    }
}
```

Archivo: **consultaClientexCodigo.php**

```
<?php include 'Conexion.php'; ?>
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">
  </head>
  <body>
    <header>
      <?php include 'encabezado.php'; ?>
      <h3>CONSULTA DE CLIENTES POR CODIGO - USANDO LA CLASE PDO</h3>
    </header>
    <section>
      <form method="POST">
        <table border="0" width="800" cellspacing="5" cellpadding="5">
          <tr>
            <td>CODIGO</td>
            <td><input type="text" name="txtCodigo" value="" /></td>
            <td><input type="submit" value="CONSULTAR" /></td>
          </tr>
        </table>
        <table border="0" width="800" cellspacing="0" cellpadding="5">
          <tr>
            <th>CODIGO</th>
            <th>CLIENTE</th>
            <th>DIRECCIÓN</th>
            <th>TELÉFONO</th>
            <th>DISTRITO</th>
            <th>EMAIL</th>
          </tr>
          <?php
            if (isset($_POST['txtCodigo'])){
              //Creando el objeto de conexión
              $objCon = new Conexion();
```

```
                //Capturando el código ingresado por el usuario
                $codigo = $_POST['txtCodigo'];

                //Script de la consulta a la tabla cliente
                $consulta="call sp_listadoclientesxcodigo('$codigo')";
                //Implementando la consulta y enviando el resultado a $rs
                $rs=$objCon->getConexion()->query($consulta);

                //Recorriendo por los registros consultados
                while($fila=$rs->fetch()){
            ?>
            <tr>
              <td><?php echo $fila[0]; ?></td>
              <td><?php echo $fila[1]; ?></td>
              <td><?php echo $fila[2]; ?></td>
              <td><?php echo $fila[3]; ?></td>
              <td><?php echo utf8_decode($fila[4]); ?></td>
              <td><?php echo $fila[5]; ?></td>
            </tr>
            <?php } //Fin del while
                 } //Fin del if
            ?>
          </table>
      </form>
    </section>
  </body>
</html>
```

Archivo: **miEstilo.css**

```
body{
    font-family: tahoma;
    font-size: 14px;
}
h3,h5{
    text-align: center;
}
table {
    margin: auto;
}
img{
    margin: auto;
    display: block;
}
td {
    border: solid 1px #006699;
    border-top: #006699;
    border-right: #006699;
    border-left: #006699;
    border-bottom: #006699;
}
table th {
background-color: #b9c9fe;
}
#centrado{
    text-align: center;
}
```

Caso desarrollado 3 **Listado de clientes por distrito**

Implementar una aplicación web con PHP que permita listar los clientes de un determinado distrito desde la base de datos VENTAS2015 usando la clase PDO, tal como se muestra en la siguiente imagen:

Imagen: consultaClientesxDistrito.php

Tener en cuenta lo siguiente:

- Implementar toda la aplicación en una carpeta llamada **consultaPDO**.
- Al iniciar la aplicación deberá mostrar la página **consultaClientesxDistrito.php**. Además de mostrar todos los distritos en el control cuadro combinado para poder seleccionar un distrito filtro.
- El botón **CONSULTAR** deberá mostrar a todos los clientes cuyo distrito sea el seleccionado desde el cuadro combinado.

El objetivo de cada archivo que contiene el proyecto es:

- **conexion.php**: Es la clase que contiene la función de conexión con el servidor y la base de datos.
- **consultaClientesxDistrito.php**: Contiene la página web que interactúa con el usuario para listar los clientes según el distrito seleccionado.
- **miEstilo.css**: Contiene los estilos usados en el archivo **consultaClientesxDistrito.php**.

Procedimientos almacenados

```
DROP PROCEDURE IF EXISTS SP_LISTADOCLIENTESXDISTRITO;
CREATE PROCEDURE SP_LISTADOCLIENTESXDISTRITO(IDD CHAR(3))
    SELECT C.ID_CLIENTE,
            CONCAT(C.NOMBRES,' ',C.PATERNO,' ',C.MATERNO) AS CLIENTE,
            C.DIRECCION,C.FONO,D.DESCRIPCION,C.EMAIL
            FROM CLIENTE C
            INNER JOIN DISTRITO D
            ON C.ID_DISTRITO=D.ID_DISTRITO
            WHERE D.ID_DISTRITO=IDD;
```

Archivo: **Conexion.php**

```
<?php
class Conexion {
    //Atributo para la cadena de conexión
    private $cn=null;
    //Función de conexión
    public function getConexion(){
    try{
      //Script de la cadena de conexión
      $cadena='mysql:host=localhost;dbname=ventas2015';

      //Implementando la cadena de conexión
    $this->cn = new PDO($cadena,'root','');
      $this->cn->setAttribute(PDO::ATTR_CASE,PDO::CASE_LOWER);

      //Retornando la conexión correcta
      return $this->cn;
     } catch(PDOException $ex){
            echo 'Error en la conexion - '.$ex->getMessage();
      }
    }
}
```

Archivo: **consultaClientesxDistrito.php**

```
<?php include 'Conexion.php'; ?>
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
    <link href="miEstilo.css" rel="stylesheet">
  </head>
  <body>
    <header>
       <?php include 'encabezado.php'; ?>
       <h3>CONSULTA DE CLIENTES POR DISTRITO - USANDO LA CLASE PDO</h3>
    </header>
    <section>
       <?php
          //Omitiendo errores de cuidado
          error_reporting(0);

          //Creando el objeto de conexión
          $objCon = new Conexion();

          //Implementando la consulta de los distritos
          $consultaD='SELECT * FROM DISTRITO';
          $rsD=$objCon->getConexion()->query($consultaD);

          //Capturamos el código del distrito seleccionado
          $distrito=$_POST['selDistrito'];

          //Script que invoca el nombre del procedimiento almacenado
          //que permite filtrar los clientes de acuerdo al distrito.
          $consultaC="call sp_listadoclientesxdistrito('$distrito')";

          //Almacenando los registros obtenidos a partir del filtro
          $rs=$objCon->getConexion()->query($consultaC);
       ?>
```

```
        <form method="POST">
          <table border="0" width="800" cellspacing="5" cellpadding="5">
            <tr>
              <td>DISTRITO</td>
              <td><select name="selDistrito">
                    <?php
                       //Hacemos que el cuadro combinado del distrito
                       //mantenga el distrito seleccionado por el usuario
                       $seleccionado='';
                       while($distritos=$rsD->fetch()){
                          if ($distritos[0]==$distrito)
                            $seleccionado=' SELECTED ';
                            else
                            $seleccionado ='';
                     ?>
                     <option value="<?php echo $distritos[0]; ?>"
                                   <?php echo $seleccionado; ?>>
                                   <?php echo $distritos[1]; ?>
                     </option>
                     <?php
                         }
                     ?>
                     </select>
                </td>
                <td><input type="submit" value="CONSULTAR" /></td>
             </tr>
          </table>
          <table border="0" width="800" cellspacing="0" cellpadding="5">
            <tr>
              <th>CODIGO</th>
              <th>CLIENTE</th>
              <th>DIRECCIÓN</th>
              <th>TELÉFONO</th>
              <th>DISTRITO</th>
              <th>EMAIL</th>
            </tr>
            <?php
              //Verificando que el usuario ha seleccionado un distrito
              if (isset($_POST['selDistrito'])){
                //Recorriendo por los registros a partir del filtro de los
                //clientes.
                while($fila=$rs->fetch()){
              ?>
              <tr>
                <td><?php echo $fila[0]; ?></td>
                <td><?php echo $fila[1]; ?></td>
                <td><?php echo $fila[2]; ?></td>
                <td><?php echo $fila[3]; ?></td>
                <td><?php echo utf8_decode($fila[4]); ?></td>
                <td><?php echo $fila[5]; ?></td>
              </tr>
              <?php } //Fin del while
                  }//Fin del if
              ?>
        </table>
      </form>
      </section>
    </body>
</html>
```

Bibliografía

McLaughlin, Brett (2013). *PHP & MySQL* 2.ª ed. United States of America: O'Reilly.

Boronczyk, Timothy (2009). *PHP6, Apache, MySQL Web Development*. Indianápolis: Wiley Publishing Inc.

Welling, Luke-Thomson, Laura (2005). *Desarrollo web con PHP y MySQL*. Madrid: Anaya S.A.

Olsson, Mikael (2010). *PHP and MySQL 24-Hour Trainer*. Indianápolis: Wiley Publishing Inc.

Impreso en los talleres gráficos de

Surquillo